TOSHIBA

PCI Connection Companion Chip TC86C001FG (GOKU-S)

Rev. 2.1

TOSHIBA CORPORATION
Semiconductor Company

The information contained herein is subject to change without notice. 021023_D

TOSHIBA is continually working to improve the quality and reliability of its products. Nevertheless, semiconductor devices in general can malfunction or fail due to their inherent electrical sensitivity and vulnerability to physical stress.
It is the responsibility of the buyer, when utilizing TOSHIBA products, to comply with the standards of safety in making a safe design for the entire system, and to avoid situations in which a malfunction or failure of such TOSHIBA products could cause loss of human life, bodily injury or damage to property.
In developing your designs, please ensure that TOSHIBA products are used within specified operating ranges as set forth in the most recent TOSHIBA products specifications. Also, please keep in mind the precautions and conditions set forth in the "Handling Guide for Semiconductor Devices," or "TOSHIBA Semiconductor Reliability Handbook" etc. 021023_A

The TOSHIBA products listed in this document are intended for usage in general electronics applications (computer, personal equipment, office equipment, measuring equipment, industrial robotics, domestic appliances, etc.).
These TOSHIBA products are neither intended nor warranted for usage in equipment that requires extraordinarily high quality and/or reliability or a malfunction or failure of which may cause loss of human life or bodily injury ("Unintended Usage"). Unintended Usage include atomic energy control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, combustion control instruments, medical instruments, all types of safety devices, etc. Unintended Usage of TOSHIBA products listed in this document shall be made at the customer's own risk. 021023_B

The products described in this document shall not be used or embedded to any downstream products of which manufacture, use and/or sale are prohibited under any applicable laws and regulations. 060106_Q

The information contained herein is presented only as a guide for the applications of our products. No responsibility is assumed by TOSHIBA for any infringements of patents or other rights of the third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of TOSHIBA or others. 021023_C

Please use this product in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances.
Toshiba assumes no liability for damage or losses occurring as a result of noncompliance with applicable laws and regulations. 060819_AF

The products described in this document may include products subject to the foreign exchange and foreign trade control laws. 060925_F

The products described in this document may contain components made in the United States and subject to export control of the U.S. authorities. Diversion contrary to the U.S. law is prohibited. 021023_H

Preface

Thank you for new or continued patronage of TOSHIBA semiconductor products. This is the 2006 edition of the user's manual for the TC86C001FG PCI Connection Companion Chip.

This databook is written so as to be accessible to engineers who may be designing a TOSHIBA TC86C001FG PCI Connection Companion Chip into their products for the first time. No prior knowledge of this device is assumed. What we offer here is basic information about the microprocessor, a discussion of the application fields in which the microprocessor is utilized, and an overview of design methods. On the other hand, the more experienced designer will find complete technical specifications for this product.

Toshiba continually updates its technical information. Your comments and suggestions concerning this and other Toshiba documents are sincerely appreciated and may be utilized in subsequent editions. For updating of the data in this manual, or for additional information about the product appearing in it, please contact your nearest Toshiba office or authorized Toshiba dealer.

October 2006

Table of Contents

Handling Precautions

1. Precautions for Semiconductor Product Use

Toshiba is continually working to improve the quality and reliability of its products. Nevertheless, semiconductor devices in general can malfunction or fail due to their inherent electrical sensitivity and vulnerability to physical stress.

It is the responsibility of the buyer, when utilizing Toshiba semiconductor products, to comply with the standards of safety in making a safe design for the entire system, and to avoid situations in which a malfunction or failure of such Toshiba semiconductor products could cause loss of human life, bodily injury or damage to property.

In developing your designs, please check the most recent product specifications and ensure that the Toshiba semiconductor products are used within the operating ranges. Also, please keep in mind the precautions and conditions set forth in this Handbook.

2. Safety Precautions

This section lists important precautions which users of semiconductor devices (and anyone else) should observe in order to avoid injury to human body and damage to property, and to ensure safe and correct use of devices. Please be sure that you understand the meanings of the labels and graphic symbols described below before you move on to the detailed descriptions of the precautions, and make every effort to observe the precautions stated.

[Explanation of Labels]

Label	Meaning
⚠DANGER	Indicates an imminently hazardous situation which will result in death or serious injury[1] if you do not followinstructions.
⚠WARNING	Indicates a potentially hazardous situation which could result in death or serious injury[1] if you do not followinstructions.
⚠CAUTION	Indicates a potentially hazardous situation which if not avoided, may result in minor injury[2], moderate injury[2], or property damage[3].

1. Serious injury includes blindness, wounds, burns (low and high temperature), electric shock, fractures, and poisoning, etc. with long-lasting effects or that require hospitalization and/or long-term hospital visits for treatment.
2. Injury includes wounds, burns, electric shock, etc. not requiring hospitalization and/or long-term hospital visits for treatment.
3. Damage includes extensive damage to machines and equipment.

[Explanation of Graphic Symbols]

Graphic Symbol	Meaning
 Prohibited	Indicates prohibited (restricted) actions. Prohibited actions are explained in or near the symbols in pictures and sentences.
 Instructions	Indicates compulsory (mandatory) actions. Compulsory actions are explained in or near the symbols in pictures and sentences.

 Caution	Indicates cautions. Cautions are explained in or near the symbols in pictures and sentences.
 Caution	Example of a caution symbol: Indicates laser beam caution.

2.1 General Precautions Regarding Semiconductor Products

⚠CAUTION	
 Prohibited	The absolute maximum ratings of a semiconductor device are a set of ratings that must not be exceeded, even for a moment. Do not exceed any of these ratings. Exceeding the rating(s) may cause the device breakdown, damage or deterioration, and may result injury by explosion or combustion.
 Prohibited	Do not insert devices in the wrong orientation or incorrectly. Make sure that the positive and negative terminals of power supplies are connected properly. Otherwise, the current or power consumption may exceed the absolute maximum rating, and exceeding the rating(s) may cause the device breakdown, damage or deterioration, and may result injury by explosion or combustion. In addition, do not use any device that is applied the current with inserting in the wrong orientation or incorrectly even just one time.
 Prohibited	Do not touch the heat sink of the device while the device is on or immediately after the device has been turned off. Heat sinks become hot. Contact to the heat sink may result in a burn.
 Prohibited	Do not touch the lead tips of a device. Some devices have leads withsharp tips. Contact to sharp tips may result in a puncture wound.
 Instructions	On the evaluation, inspection or test, be sure to connect the test equipment's electrodes or probes to the pins of the device before turning the power on. When you have finished, discharge any electrical charge remaining in the device. Insufficient connection, wrong power-on timing or incomplete discharge may cause electric shock, resulting in injury.
 Instructions	Check that there is no electrical leakage before grounding measuring equipment or a solder iron. Electrical leakage may cause the device you are testing or soldering to electrically break down or may cause electric shock.
 Instructions	Always wear safety glasses when cutting the leads of a device with clippers or a similar tool. Failure to do so may result in eye damage from the small shavings that fly off the cut ends.

2.2 Precautions Specific to Product Group

2.2.1 Optical Semiconductor Devices

⚠DANGER	
 Prohibited	When a semiconductor laser is operating, do not look into the laser beam or look through the optical system. Doing so may damage your eyesight or, in the worst case, cause blindness. When inspecting the optical characteristics of the laser using laser protective glasses, be sure the glasses comply with JISC6802.
⚠WARNING	
 Prohibited	Do notapply voltage or current into the LED device that exceeds the device's absolute maximum rating. With resin-packaged LED devices in particular, excessive voltage or current may cause the package resin to explode, scattering resin fragments and may result injury.
 Instructions	When evaluating and testing the dielectric strength voltage of a photocoupler, use equipment that can shut off the supply voltage if a leakage current exceeding 100μA is detected. Failure to do so may result in the continuous flow of a large short-circuit current, causing the device to explode or combust, resulting in fire or injury.
 Instructions	When designing a semiconductor laser, use the built-in or another light-receiving element to stabilize optical output so as to ensure that laser beams exceeding the laser's rated optical output are not emitted. If this stabilization function does not work properly and the rated output is exceeded, not only the device may break down but also injury may result by the laser beam.

2.2.2 Power Device

⚠DANGER	
 Prohibited	Do not touch a power device while it is on or after it has been turned off and all remaining electrical charge has not yet been discharged. Touching power element with electrical charge may cause electric shock, resulting in death or serious injury.
 Instructions	When evaluating, inspecting or testing a device, be sure to connect all of the test equipment's electrodes or probes before turning the power on. When you have finished, discharge any electrical charge remaining in the device. Connecting the electrodes or probes with the power on may cause electric shock, resulting in death or serious injury.

⚠WARNING	
Prohibited	Do not use a device under conditions that exceed its absolute maximum ratings (such as current, voltage, safe operation range, temperature.). Using device with exceeded its absolute maximum ratings may cause the device to break down, causing a short-circuit current, which may in turn cause it to explode or combust, resulting in fire or injury.
Instructions	Use a unit which can detect short-circuit currents and shut off the power supply if a short-circuit occurs. If the power supply is not shut off, a large short-circuit current will flow continuously, causing the device to explode or combust, resulting in fire or injury.
Instructions	Design the case for enclosing your system taking into consideration the prevention of flying debris in the event the device explodes or combusts. Flying debris may cause injury.
Instructions	When conducting an evaluation, inspection or test, use protective safety tools such as device covers. The device may explode or combust due to excessive stress in the event of breakdown or arc discharge between the electrode and ground potential, resulting in fire or injury.
Instructions	Design your product so that it is used with all metal areas (other than electrodes and terminals) grounded. Even with products where the device's electrodes and metal casings are insulated, electrostatic capacitance in the product may cause the electrostatic potential in the casing to rise. Insulation deterioration or breakdown may cause a high voltage to be applied to the casing, causing electric shock when touched, resulting in death or serious injury.
Instructions	When designing the heat radiation and safety features of a system incorporating Schottky barrier diodes and high-speed rectifiers, take into consideration the device's forward and reverse losses. The reverse current in these devices is greater than that in ordinary rectifiers. If the operating environment is severe (such as high temperature, high voltage), the device's reverse loss may increase, causing a short-circuit current and subsequently explosion or combustion, resulting in fire or injury.
Instructions	Be sure to design the product so that, except when the main circuits of the device are active, the main circuits will be inactive when electricity is conducted to control circuits. Malfunction of the device may cause a serious accident or injury.
⚠CAUTION	
Instructions	When conducting an evaluation, inspection or test, either wear protective gloves or wait until the device cools down prior to handling. Devices become hot when operated. Even after the power supply has been turned off, residual heat may cause a burn when the device is touched.

2.2.3 Application Specific Standard Products and General-Purpose Linear ICs

⚠CAUTION	
 Instructions	Use an appropriate power supply fuse to ensure that a large current does not continuously flow in case of over current and/or IC failure. The IC will fully break down when used under conditions that exceed its absolute maximum ratings, when the wiring is routed improperly or when an abnormal pulse noise occurs from the wiring or load, causing a large current to continuously flow and the breakdown can lead smoke or ignition. To minimize the effects of the flow of a large current in case of breakdown, appropriate settings, such as fuse capacity, fusing time and insertion circuit location, are required.
 Instructions	If your design includes an inductive load such as a motor coil, incorporate a protection circuit into the design to prevent device malfunction or breakdown caused by the current resulting from the inrush current at power ON or the negative current resulting from the back electromotive force at power OFF. For details on how to connect a protection circuit such as a current limiting resistor or back electromotive force adsorption diode, refer to individual IC datasheets or the IC databook. IC breakdown may cause injury, smoke or ignition.
 Instructions	Use a stable power supply with ICs with built-in protection functions. If the power supply is unstable, the protection function may not operate, causing IC breakdown. IC breakdown may cause injury, smoke or ignition.
 Instructions	Carefully select external components (such as inputs and negative feedback capacitors) and load components (such as speakers), for example, power amp and regulator. If there is a large amount of leakage current such as input or negative feedback condenser, the IC output DC voltage will increase. If this output voltage is connected to a speaker with low input withstand voltage, over current or IC failure can cause smoke or ignition. (The over current can cause smoke or ignition from the IC itself.) In particular, please pay attention when using a Bridge Tied Load (BTL) connection type IC that inputs output DC voltage to a speaker directly.

2.2.4 Memory Card Products

⚠WARNING	
Prohibited	Keep out of reach of small children. Accidental swallowing may cause suffocation or injury. Contact a doctor immediately if you suspect a child has swallowed the Product.

⚠CAUTION	
Prohibited	Do not directly touch the interface pins, put them in contact with metal, strike them with hard objects, or cause them to short. Do not expose to static electricity.
Prohibited	Do not bend, apply strong force to, drop, expose to strong impact or lay heavy objects on top of the Product.
Prohibited	Do not put the Product in the back pocket, etc. of trousers. It may break when you sit down or exert strong force on it in other ways.
Prohibited	Do not disassemble or modify the Product. This may cause electric shock, damage to the Product, or fire.
Prohibited	Do not expose the Product to moisture. Do not use, store, or place in humid locations or expose to water. Do not expose the Product to excessive heat or cold. Do not use, store, or place in direct sunlight, inside a hot car, near fire or sources of heat or flame, such as a stove. Do not use, store, or place the Product near an air-conditioner outlet. Do not expose the Product to dust, strong magnetic fields, or corrosive chemicals or gas.
Prohibited	Avoid sudden temperature changes which could cause condensation.

⚠CAUTION	
 Prohibited	(Applicable only to mini SD) While writing data to or reading data from the Product, do not turn off the power, remove the Product or the miniSD adapter from the device, or permit the Product or device to be shaken or impacted.
 Prohibited	(Applicable only to mini SD) Do not insert, remove or change the Product while the miniSD adapter is still inserted into a device. This may cause product failure or the destruction or loss of data.
 Prohibited	(Applicable only to mini SD) Do not insert into a device the miniSD adapter that does not contain the Product. This may cause improper function on the device.
 Prohibited	(Applicable only to mini SD) Do not insert into the miniSD adapter a memory card other than the Product or other foreign object. This may cause product failure.
 Prohibited	While writing data to or reading data from the Product, do not turn off the power, remove the Product from the device, or permit the Product or device to be shaken or impacted.
 Prohibited	(Applicable only to USB flash memory) Performing the following operations with the Product connected to your PC may cause improper function on your PC. - Booting - Rebooting - Resuming from standby or suspended mode Please perform these operations after removing the Product from your computer.
 Instructions	The Product comes pre-formatted. When formatting, all data stored in the Product will be lost, so make sure to back up the data on the Product. Please use functions relevant to the Product. (Applicable to SD, mini SD and compact flash) Formatting with other devices, for example PCs, may cause problems such as the inability to read, write, or delete data.
 Instructions	(Applicable only to USB flash memory) Please take note of the following when formatting the Product using Windows XP or Windows 2000. · Log in as a user with administrator access rights. The Product cannot be formatted while logged in as a user with limited access rights. · Format the Product as FAT. It may not format properly as NTFS or FAT32.
 Instructions	When removing the Product from the slot or the USB port of a PC, first follow the removal (stop) procedure for your operating system.

⚠CAUTION	
 Instructions	Refer to your device's manual to learn how to insert and remove the Product.
 Instructions	Insert the Product firmly in the correct orientation. The Product will not operate correctly if it is inserted in an incorrect orientation or not inserted all the way.
 Instructions	Always use the Product with the interface pins and connector in a clean state. When cleaning the Product, use a soft, dry cloth.
 Instructions	(Applicable only to miniSD) Always use the miniSD adapter when using the Product with a standard SD memory card device. Directly inserting the Product into a standard SD memory card device may cause improper function on the device.

3. General Safety Precautions and Usage Considerations

This section provides information that will help you gain a better understanding of semiconductor devices so as to ensure device safety, quality and reliability.

3.1 From Incoming to Shipping

3.1.1 Electrostatic Discharge (ESD)

When handling individual devices, be sure that the environment is protected against static electricity. Operators should wear anti-static clothing. In addition, containers and other objects that come in direct contact with devices should be made of materials that do not produce static electricity that would cause damage. Please follow the precautions below. This is particularly important for those devices marked “Be careful of static.”

3.1.1.1 Work Environment Control

(1) When humidity decreases, static electricity readily occurs due to friction. Taking into consideration the fact that moisture-proof-packed products absorb moisture after unpacking, the recommended humidity is 40 to 60%.

(2) Be sure that all equipment such as jigs and tools installed in the work area are grounded.

(3) Place a conductive mat over the floor of the work area or take other measure to ensure that the floor is protected against static electricity and is grounded to the earth. (Resistance between surface and ground: 1 x $10^9\Omega$ or less)

(4) Place a conductive mat over the surface of worktables to ensure that the tables are grounded. (Resistance between surface and ground: 7.5 x 10^5 to 1 x $10^9\Omega$) Do not construct worktable surfaces of metallic materials. Metallic materials are low in resistance, allowing rapid discharge when a charged device comes in contact with them directly.

(5) Observe the following when using automated equipment:

(a) When picking up a device with a vacuum, use conductive rubber in sections which come into contact with the device surface to protect against electrostatic charge.

(b) Avoid friction on the device surface to the extent possible. If rubbing is unavoidable due to the device’s mechanical structure, minimize the friction plane or use material with a small friction coefficient and low electrical resistance. Also, prevent electrostatic charge by using an ionizer.

(c) Use a material which dissipates static electricity in sections which come in contact with device leads or terminals.

(d) Ensure that no statically charged bodies (such as work clothes or the human body) come in contact with the devices.

(e) Use a tape carrier that employs a low-resistance material on sections that come in contact with

electrical machinery.

(f) Make sure that jigs and tools used in the manufacturing process do not touch the devices.

(g) In processing associated with package electrostatic charge, use an ionizer to neutralize the ions in the ambient environment.

(6) Make sure that CRT displays in the work area are protected against static charge by employing a filter, for example. Avoid turning displays on and off to the extent possible. Neglecting to do so can cause electrostatic induction in devices.

(7) Periodically measure the charged potential of devices, systems and fixtures located in the work area to ensure that the area is free of any charge.

(8) Ensure that the work chairs are protected by a conductive cover and grounded to the floor by conductive castors. (Resistance between seat surface and ground: 1 x $10^{10}\Omega$ or less)

(9) Install anti-static mats on storage shelf surfaces and ground the mat surface. (Resistance between surface and ground: 7.5 x 10^5 to 1 x $10^9\Omega$)

(10) For device transport and temporary storage, use containers (boxes, jigs or bags) that are made of a material which does not produce static electricity that could damage the device.

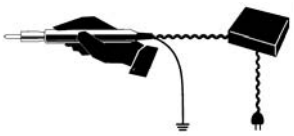
(11) Make sure that cart surfaces which come in contact with product packaging are made of materials which conduct static electricity, and ground the cart surfaces to the floor surface using conductive castors.

(12) In static electricity control areas, install anti-static dedicated ground wires. Use a transmission line circuit ground wire [Type D (previous Class C) or above], or a trunk line ground wire. In addition, separate and ground the various devices individually.

3.1.1.2 Work Environment Control

(1) Operators must wear anti-static clothing and conductive shoes (or a toe or heal strap).

(2) Operators must wear a wrist strap grounded to earth via a resistor. (Resistance between surface and earth when worn: 7.5 x 10^5 to 3.5 x $10^7\Omega$)



(3) Soldering irons must be grounded from the iron tip to earth, and must be used at low voltages (6 to 24V).

(4) If the tweezers you use are likely to touch the device terminals, use anti-static tweezers. Do not use metallic tweezers since they are low in resistance and may cause rapid discharge when a charged device comes in contact with them.

When using a vacuum tweezers, attach a conductive chucking pat to the tip, and connect it to a dedicated anti-static ground. In addition, follow the manufacturer's methods of use and maintenance.

(5) Do not place devices or their containers near sources of strong electrical fields (such as above a CRT).

(6) Place boards with mounted devices in anti-static board containers separated from one another, and do not stack them directly on top of one another. Stacking them directly on top of one another may cause frictional charge or discharge.

(7) Ensure, to the extent possible, that any articles (such as clipboards) which are brought to a static electricity control area are constructed of anti-static materials.

(8) When the human body is to come in direct contact with a device, wear anti-static finger covers or gloves.

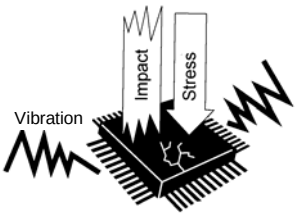
(9) The material of equipment safety covers located near devices should have a resistance rating of 1 x $10^9\Omega$ or less.

(10) If a wrist strap cannot be used and there is a possibility of imparting friction to devices, use an ionizer.

(11) The transport film used in tape carrier products is manufactured from materials in which static electricity readily builds up. When using these products, use an ionizer to prevent the film from being charged. Also, to ensure than no static electricity will be applied to the copper foil area, take measures to prevent electrostatic discharge failure of peripheral equipment.

3.1.2 Vibration, Impact and Stress

Handle devices and packaging with care. Dropping or applying impact to devices or packaging causes device damage. Ensure that devices and packaging are not subjected to mechanical vibration or impact to the extent possible. Hollow canister-type devices and ceramic sealed devices contain unsecured wires, making them more susceptible to vibration and impact than plastic sealed devices.



When a location such as a soldered area, connecting area or top surface of a device is subjected to vibration, impact or stress in actual equipment, bonding fault or device destruction may result. Therefore, be sure to keep this in mind at the time of structural design. If a device is subject to especially strong vibration, impact or stress, the package or chip may crack. If stress is applied to a semiconductor chip through the package, changes in the resistance of the chip may result due to piezoelectric effects, resulting in fluctuation in element characteristics. Furthermore, if a stress that does not instantly result in damage is applied continually for a long period of time, product deformation may result, causing defects such as disconnection or element failure. Thus, at the time of structural design, carefully consider vibration, impact and stress.

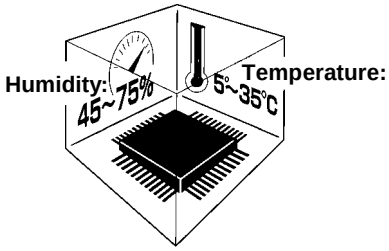
3.2 Storage

3.2.1 General Packaged Products

(1) Avoid storage locations where devices may be exposed to moisture or direct sunlight.

(2) Follow the precautions printed on the packing label of the device for transportation and storage.

(3) Keep the storage location temperature and humidity within a range of 5°C to 35°C and 45% to 75%, respectively.



(4) Do not store the products in locations with poisonous gases (especially corrosive gases) or in dusty conditions.

(5) Store the products in locations with minimal temperature fluctuations. Rapid temperature changes during storage can cause condensation, resulting in lead oxidation or corrosion, which will deteriorate the solderability of the leads.

(6) When restoring devices after removal from their packing, use anti-static containers.

(7) Do not allow loads to be applied directly to devices while they are in storage.

(8) If devices have been stored for more than two years under normal storage conditions, it is recommended that you check the leads for ease of soldering prior to use.

3.2.2 Moisture-Proof Packing

Moisture-proof packing should be used while taking into careful consideration the handling methods specified for each packing type. If the specified procedures are not followed, the quality and reliability of the devices may be deteriorated. This section describes the general precautions for handling moisture-proof packing. Since the details may differ from device to device, refer to the individual standards or databooks during handling.

3.2.2.1 Moisture-Proof Packing General Precautions

Follow the precautions printed on the packing label of the device for transportation and storage.

For chip products, follow the individual specifications.

(1) Do not toss or drop device packing. The aluminum laminated bag may be damaged, resulting in a loss in airtightness.

(2) Keep the storage environment at 5°C to 30°C, and the relative humidity at 90% or less. Use devices within 12 months of the date marked on the package seal.

(3) If the 12-month storage period has been exceeded, or if the 30% humidity indicator is pink when the package is opened, remove any moisture under the conditions described in the table below. The effective usage period without moisture removal after the packing has been opened and the product has been stored at 5°C to 30°C and a relative humidity of 60% is listed on the moisture-proof package. If the effective usage period has been exceeded, or if the packing has been stored in a high-humidity environment or an environment that produces condensation, remove any existing moisture.

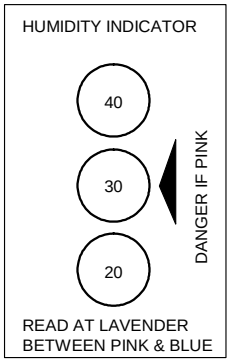
Packing Type	Moisture Removal Applicability/Procedure
Tray	If the packing indicates a “Heatproof” or temperature label, bake at 125°C for 20 hours. (Some devices may require a different amount of time.) If the packing does not indicate a “Heatproof” or temperature label, transfer the devices to an anti-static container that bears a “Heatproof” or temperature label and then bake. The moisture-proof package itself is not heat resistance. Be sure to remove the devices from the package prior to baking.
Magazine	Transfer devices to antistatic containers bearing the “Heatproof” or temperature label, and then bake at 125°C for 20 hours. (Some devices may require a different amount of time.) The moisture-proof package itself is not heat resistance. Be sure to remove the devices from the package prior to baking.
Tape	Transfer devices to antistatic containers bearing the “Heatproof” or temperature label, and then bake at 125°C for 20 hours. (Some devices may require a different amount of time.) The moisture-proof package itself is not heat resistance. Be sure to remove the devices from the package prior to baking.

(4) When removing the moisture from the devices, protect the devices from breakdown from static electricity.

(5) Moisture indicators (for your reference)

Moisture indicators detect the approximate ambient humidity level at a standard temperature of 25°C.

Figure 3.1 shows a 3-point indicator.



3-point indicator

Figure 3.1 Humidity Indicator

(6) Do not allow loads to be applied directly to devices while they are in storage.

3.3 Design

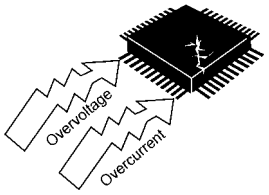
To achieve the reliability required by an electronic device or system, it is important not only to use the semiconductor device in accordance with specified absolute maximum ratings and operating ranges, but also to consider the environment in which the equipment will be used, including factors such as the ambient temperature, transient noise and current surges, as well as mounting conditions which affect semiconductor device reliability. This section describes general design precautions. Be sure to refer to the individual ratings of each product at the time of design.

3.3.1 Absolute Maximum Ratings

⚠CAUTION The absolute maximum ratings of a semiconductor device are a set of ratings that must not be exceeded, even for a moment. Do not exceed any of these ratings.

Exceeding the rating(s) may cause device breakdown, damage or deterioration, and may result injury by explosion or combustion.

If the voltage or current on any pin exceeds the absolute maximum rating, the overvoltage or overcurrent causes the device's internal circuitry to deteriorate. In extreme cases, heat generated in internal circuitry can fuse wiring or cause the semiconductor chip to break down. If the storage or operating temperature exceeds the absolute maximum rating, the device internal circuitry may deteriorate and the bonded areas may open or the package airtightness may deteriorate due to the differences between the thermal expansion coefficients of the materials from which the device is constructed.



Although absolute maximum ratings differ from product to product, they essentially concern the voltage and current at each pin, the allowable power dissipation, the connecting area temperatures, and storage temperatures. Note that the term "maximum rating" which appears in semiconductor technical datasheets and the like refers to "absolute maximum rating."

3.3.2 Operating Range

The operating range is the range of conditions necessary for the device to operate as specified in individual technical datasheets and databooks. Care must be exercised in the design of the equipment. If a device is used under conditions that do not exceed absolute maximum ratings but exceed the operating range, the specifications related to device operation and electrical characteristics may not be met, resulting in a decrease in reliability.

If greater reliability is required, derate the device's operating ranges for voltage, current, power and temperature before use.

3.3.3 Derating

The term “derating” refers to ensuring greater device reliability by setting operating ranges reduced from rated values and taking into consideration factors such as current surges and noise.

While derating generally applies to electrical stresses such as voltage, current and power, and environmental stresses such as ambient temperature and humidity, it differs from application to application. Refer to the individual technical datasheets available for each product. Power devices in particular require heat sink consideration as well since the level of derating greatly affects reliability.

For your reference, details are provided in the appendix. Be sure to read the appendix carefully.

3.3.4 Unused Pins

If unused pins are left open, some devices exhibit input instability, resulting in faulty operation such as a sudden increase in current consumption. In addition, if unused output pins on a device are connected to the power supply, GND or other output pin, the IC may malfunction or break down.

Since the treatment of unused input and output pins differs for each product and pin, please follow the directions in the individual technical datasheets and databooks.

CMOS logic IC inputs, for example, have extremely high impedance. If an input pin is left open, it can readily pick up noise and become unstable. In this case, if the input reaches an intermediate level, both the P-channel and N-channel transistors will become conductive, allowing unnecessary power supply current to flow. It is therefore necessary to ensure that the unused input gates of a device are connected to the power supply pin or ground (GND) pin of the same device. For treatment of heat sink pins, refer to the individual technical datasheets and databooks.

3.3.5 Latch-up

Semiconductor devices sometimes transition to an inherent condition referred to as “latch-up.” This condition mainly occurs in CMOS devices. This happens when a parasitic PN-PN junction (thyristor structure) built in the device itself is turned on, causing a large current to flow between the power supply voltage and GND, eventually causing the device to break down.

Latch-up occurs when the voltage impressed on an input or output pin exceeds the rated value, causing a large current to flow in the internal element, or when the voltage impressed on the power supply voltage pin exceeds its rated value, forcing the internal element to breakdown. Once the element falls into the latch-up state, even though the excess voltage may have been applied only for an instant, the large current continues to flow between the power supply voltage and GND, potentially causing device explosion or combustion. To avoid this problem, observe the following:

(1) Do not allow the voltage levels on the input and output pins to rise above the power supply voltage or decrease below GND. Consider the timing during power supply activation as well.

(2) Do not allow any abnormal noises to be applied to the device.

(3) Set the electrical potential of unused input pins to the power supply voltage or GND.

(4) Do not create an output short.

3.3.6 Input/Output Protection

Wired-AND configurations in which outputs are connected together directly cannot be used since the outputs short-circuit with the configurations. Outputs should, of course, never be connected to the power supply voltage or GND. In addition, products with tri-state outputs can undergo IC deterioration if a shorted output current continues for a long period of time. Design the circuit so that the tri-state outputs will not be enabled simultaneously.

3.3.7 Load Capacitance

Certain devices exhibit an increase in delay times and a large charging and discharging current if a large load capacitance is connected, resulting in noise. In addition, since outputs are shorted for a long period of time, wiring can become fused. Use the load capacitance recommended for each product.

3.3.8 Thermal Design

The failure rate of semiconductor devices largely increases as the operating temperatures increase. As shown in Figure 3.2, the thermal stress applied to device internal circuitry is the sum of the ambient temperature and the temperature rise caused by the power consumption of the device. For thermal design, therefore, refer to the precautions stated in individual technical datasheets and databooks.

To achieve even higher reliability, take into consideration the following thermal design points:

(1) Conduct studies to ensure that the ambient temperature (Ta) is maintained as low as possible, avoiding the effects of heat generation from the surrounding area.

(2) If the device's dynamic power consumption is relatively large, conduct studies regarding use of forced air-cooling, circuit board composed of low thermal resistance material, and heat sinks. Such measures can lower the thermal resistance of the package.

(3) Derate the device's absolute maximum ratings to minimize thermal stress from power consumption.

$\theta ja = \theta jc + \theta ca$

$\theta ja = (Tj - Ta)/P$

$\theta jc = (Tj - Tc)/P$

$\theta ca = (Tc - Ta)/P$

where, θja: Thermal resistance between junction and ambient air (°C/W)

θjc: Thermal resistance between junction and package surface, or internal thermal resistance (°C/W)

θca: Thermal resistance between package surface and ambient air, or external thermal resistance (°C/W)

Tj: Junction temperature or chip temperature (°C)

Tc: Package surface temperature or case temperature (°C)

Ta: Ambient temperature (°C)

P: Power consumption (W)

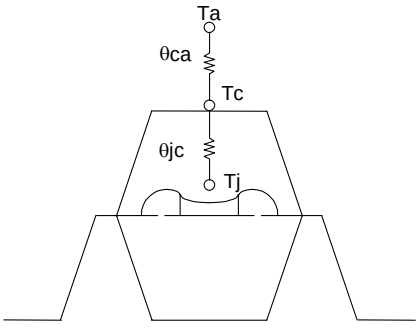


Figure 3.2 Thermal Resistance of Package

3.3.9 Interfacing

When connecting devices with different input and output voltage levels, make sure that the input voltage (V_{IL}/V_{IH}) and output voltage (V_{OL}/V_{OH}) levels match. Otherwise, the devices may malfunction. In addition, when connecting devices with different power supply voltages, such as in a dual power supply system, device breakdown may result if the power-on and power-off sequences are incorrect. For device interface details, refer to the individual technical datasheets and databooks. In addition, if you have any questions about interfacing, contact your nearest Toshiba office or distributor.

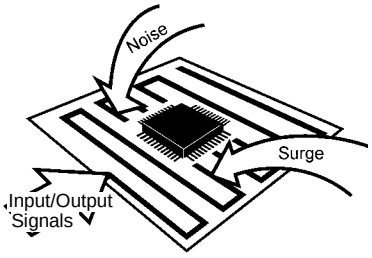
3.3.10 Decoupling

Spike currents generated during switching can cause power supply voltage and GND voltage levels to fluctuate, causing ringing in the output waveform or a delay in the response speed. (The power supply and GND wiring impedance is normally 50 to 100Ω.) For this reason, the impedance of the power supply lines with respect to high frequencies must be kept low. Specifically, this is ideally accomplished by routing thick and short power supply and GND lines and by inserting decoupling capacitors (of approximately 0.01 to 1μF) as high-frequency filters between the power supply and GND into each required location on the circuit board.

For low-frequency filtering, it is appropriate to insert a 10 to 100μF capacitor in each circuit board. However, conversely if the capacitance is excessively large (such as 1000μF), latch-up may result. An appropriate capacitance value is therefore required.

On the other hand, in the case of high-speed logic ICs, noise is caused by reflection, crosstalk or common power supply impedance. Reflections cause increased signal delay, ringing, overshoot and undershoot, thereby reducing the device's noise margin. One effective wiring measure for preventing reflections is to reduce the wiring length by increasing the mounting density so as to lower the wiring inductance (L) and capacitance (C). This measure, however, also requires consideration with regard to crosstalk between wires. In actual pattern design, both of these factors must be considered.

3.3.11 External Noise



When externally induced noise or surges are applied to a printed circuit board with long I/O signals or signal lines, malfunction may result, depending on the device. To protect against noise, protective measures against surges must be taken such as lowering the impedance of the signal line or inserting a noise-canceling circuit.

For details of required protection, refer to individual technical datasheets and databooks.

3.3.12 Electromagnetic Interference

Radio and TV reception problems have increased in recent years as a result of increased electromagnetic interference radiated from electrical and electronic equipment. To use radio waves effectively and to maintain the quality of radio communications, each country has defined limitations for the amount of electromagnetic interference which can be generated by designated devices.

The types of electromagnetic interference include noise propagated through power supply and telephone lines, and noise from direct electromagnetic waves radiated from equipment. Different measurement methods and corrective actions are used for each type.

Difficulties in countering electromagnetic interference derive from the fact that there is no means for calculating at the design stage the strength of the electromagnetic waves produced from each component in a piece of equipment. As a result, it is after the prototype equipment has been completed that measurements are taken using dedicated instruments to determine for the first time the strength of the electromagnetic interference. Yet it is possible during system design to incorporate measures for the prevention of electromagnetic interference which can facilitate corrective action after design completion. One effective method, for example, is to design the product with several shielding options, and then select the optimum shielding method based on the results of the measurements subsequently taken.

3.3.13 Peripheral Circuits

In many cases semiconductor devices are used with peripheral circuits and components. The input and output signal voltages and currents in these circuits must be designed to match the specifications of the device, taking into consideration the factors below.

(1) Input voltages and currents that are not appropriate with respect to the input pins may cause malfunction. Some devices contain pull-up or pull-down resistors, depending on specifications. Design your system taking into account the required voltage and current.

(2) The output pins on a device have a predetermined external circuit drive capability. If a drive capability exceeding this value is required, either insert a compensating circuit or take that fact into account when selecting components for use in external circuits

3.3.14 Safety Standards

Each country and region has established safety standards which must be observed. These safety standards sometimes include requirements for quality certification systems and insulation design standards. The safety standards of the respective countries and regions must be taken fully into account to ensure compliant device selection and design.

3.3.15 Other

(1) When designing a system, incorporate fail-safe and other measures according to system application. In addition, debug the system under actual mounting conditions.

(2) If a plastic package device is placed in a strong electric field, surface leakage may occur due to charge-up, resulting in malfunction. When using such a device in a strong electric field, take measures by, for example, protecting the package surface with a conductive shield.

(3) With some memory devices and microcomputers, attention is required at power on or reset release. To ensure that your design is device appropriate, refer to the individual technical datasheets and databooks.

(4) Design the casing so as to ensure that no conductive material (such as a metal pin) can drop from an external source onto a terminal of a mounted device, causing a short.

3.4 Inspection, Testing and Evaluation

3.4.1 Grounding

⚠CAUTION

Check that there is no electrical leakage before grounding measuring equipment or a solder iron.
Electrical leakage may cause the device you are testing or soldering to electrically break down or may cause electric shock.

3.4.2 Inspection Sequence

⚠CAUTION

[1] Do not insert devices in the wrong orientation or incorrectly. Make sure that the positive and negative terminals of power supplies are connected properly. Otherwise, the current or power consumption may exceed the absolute maximum rating, and exceeding the rating(s) may cause the device breakdown, damage or deterioration, and may result injury by explosion or combustion. In addition, do not use any device that is applied the current with inserting in the wrong orientation or incorrectly even just one time.

[2] On the evaluation, inspection or test using AC power with a peak value of 42.4V or DC power exceeding 60V, be sure to connect the electrodes or probes of the testing equipment before activating the power. When you have finished, discharge any electrical charge remaining in the device. Insufficient connection, wrong power-on timing or incomplete discharge may cause electric shock, resulting in injury.

(1) Apply voltage to the device after inserting it into the test jig. At this time, observe the power supply activation or shutdown standards, if existent.

(2) After test completion, be sure that the voltage applied to the device is off before removing the device from the test jig. Removing the device with the power supply on can cause device deterioration or breakdown.

(3) Make sure that no surge voltages from the measuring equipment are applied to the device.

(4) The chips in tape carrier packages (TCPs) are LSI chips and therefore exposed. During inspection, be careful not to crack or scratch the chip.
Electrical contact may also cause chip failure. Therefore make sure that nothing comes into electrical contact with the chip.

3.5 Mounting

There are two types of device packages: lead insertion and surface mount. The items that affect reliability during circuit board mounting include contamination by flux and thermal stress during the soldering process. With surface-mount devices in particular, the most significant problem is thermal stress from solder reflow, when the entire package is subjected to heat. In addition, the mounting method differs according to factors such as chip size and frame design, even for the same package type. For details, refer to the individual technical datasheets and databooks for each device.

When a location such as a soldered area, connecting area or top surface of a device is subjected to vibration, impact or in actual equipment, bonding fault or device destruction may result. Therefore, be sure to keep this in mind at the time of mounting. If a device is subject to especially strong vibration, impact or stress, the package or chip may crack. Thus, at the time of mounting, carefully consider vibration, impact and stress.

3.5.1 Lead Forming

⚠CAUTION

[1] Always wear safety glasses when cutting the leads of a device with clippers or a similar tool. Failure to do so may result in eye damage from the small shavings that fly off the cut ends.

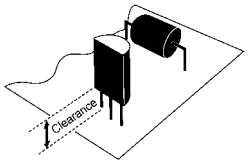
[2] Do not touch the lead tips of a device.
Some devices have leads withsharp tips. Contact to sharp tips may result in a puncture wound.

Semiconductor devices sometimes undergo a process in which the leads are cut and formed before the devices are installed on a printed circuit board. If abnormal stress is applied to the interior of a device during this process, mechanical breakdown or reliability deterioration may result. This is attributable mainly to the relative stress applied between the device itself and the lead, and can result in internal lead damage, adhesive property deterioration and sealant breakdown. Observe the following precautions during the lead-forming process.
(This does not apply to surface-mount devices.)

(1) Lead insertion hole intervals on the printed circuit board should be designed using the same dimension standard as that for the lead interval of the device.

(2) If the lead insertion hole intervals on the printed circuit board do not match the lead interval of the device, do not forcibly insert the device.

(3) For the minimum dimension between a device and printed circuit board, refer to the individual technical datasheets and databooks. When necessary, create space when forming the device's leads. Do not use the spacers for raising devices above the surface of the printed circuit board during soldering. These spacers may continue to expand due to heat even after the solder has solidified, sometimes applying a great amount of stress to the device.



(4) Observe the following when forming the leads of a device:
   (a) When bending a lead, secure the lead at the end of the bending section near the package to ensure that mechanical stress is not applied to the device. Also, do not repeatedly bend or stretch a lead at the same location.
   (b) Do not damage the lead during lead forming.
   (c) Following any other precautions specified in the individual technical datasheets or databooks.

3.5.2 Socket Mounting

(1) When socket-mounting devices on a printed circuit board, use sockets that match the package.
(2) Use sockets with contacts that have the appropriate contact pressure. If the contact pressure is insufficient, the contact may become poor when the device is repeatedly inserted and removed. If the contact pressure is too high, the device leads may bend or become damaged when they are inserted into or removed from the socket.
(3) When soldering sockets to the printed circuit board, use sockets designed to prevent flux from penetrating the contacts and to allow flux to be completely cleaned off.
(4) Ensure that the coating agent applied to the printed circuit board for moisture-proofing does not adhere to the socket contacts.
(5) If the leads are severely bent when inserted into or removed from a socket and you want to repair the leads and continue using the device, repair the leads once only. Do not use devices whose leads have been corrected multiple times.
(6) If external vibration will be applied to a printed circuit board with devices mounted on it, use sockets with strong contact pressure so as to prevent vibration between the devices and sockets.

3.5.3 Lead(Pb)-Free / Lead(Pb)-Free Finish* Soldering Temperature Profile

Perform soldering following the methods and conditions described in the individual technical datasheets and databooks for the device used. The soldering method, temperature and time may be restricted, depending on the device. All soldering temperature profiles and conditions described in the mounting methods below are representative. The profiles and conditions vary from product to product. Therefore, mount the product after first confirming the information described in the individual technical datasheets and databooks with the customer.

For details regarding lead(Pb) soldering, please contact your nearest Toshiba office or distributor.

*Toshiba Semiconductor Company defines capitalized "Lead(Pb)-Free" products as those Containing no more than 0.1 percent lead(Pb) by weight in homogeneous materials. This does not mean that Toshiba Semiconductor products labeled "Lead(Pb)-Free" are entirely free of

lead(Pb). In addition to Lead(Pb)-Free, Toshiba Semiconductor Company will offer products that have Lead(Pb)-Free terminals, which will be referred to as “Lead(Pb)-Free Finish.” The Lead(Pb)-Free Finish products may Contain greater than 0.1 percent lead(Pb) by weight in homogeneous materials in portions of the product other than the terminals (based on the exemption(s) in the RoHS Directive), for example, in internal solder used to connect the semiconductor silicon to the package. This does not mean that Toshiba Semiconductor products that are labeled "Lead(Pb)-Free Finish" have terminals that are entirely free of lead(Pb). Furthermore, the expressions “Lead(Pb)-Free” and “Lead(Pb)-Free Finish” will be changed in package labeling as the like below from April 2006.

<<Examples of correspondence with the existing lead (Pb)-free markings>>

[Lead (Pb)-free products]: Lead (Pb)-Free -> [[G]]/RoHS COMPATIBLE

[Lead (Pb)-free finish products]: Lead(Pb)-Free Finish -> [[G]]/RoHS [[Pb]]

3.5.3.1 Using a Soldering Iron

Complete soldering within 10 seconds for lead temperatures of up to 260°C, or within 3 seconds for lead temperatures of up to 350°C.

3.5.3.2 Using Infrared Reflow

(1) It is recommended the top and bottom heating method with long or medium infrared rays. (See Figure 3.3.)

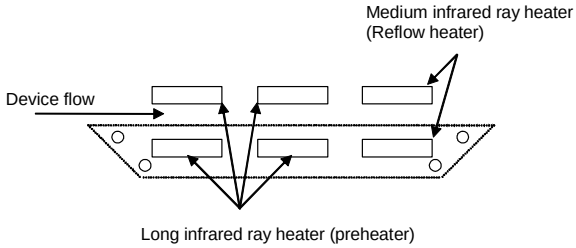


Figure 3.3 Top and Bottom Heating Method with Long or Medium Infrared Rays

(2) Complete the infrared ray reflow process with a maximum package surface temperature of 260°C, within 30 to 50 seconds when a package surface temperature is 230°C or higher.

(3) Refer to Figure 3.4 for an example of a temperature profile.

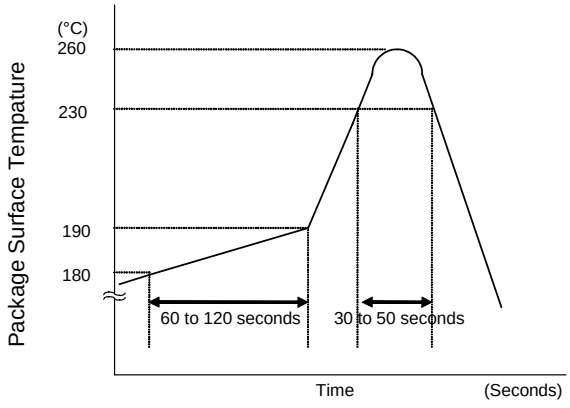


Figure 3.4 Example of Temperature Profile

This profile is based on the device's maximum heat resistance guaranteed value.

Set the preheat temperature/heating temperature to the optimum temperature corresponding to the solder paste type used by the customer within the above-described profile.

3.5.3.3 Using Hot Air Reflow

(1) Complete hot air reflow with a maximum package surface temperature of 260°C, within 30 to 50 seconds when a package surface temperature is 230°C or higher.

(2) For an example of a temperature profile, refer to Figure 3.4 in Section 3.5.3.2 (3) above.

3.5.3.4 Using Solder Flow/Dip

(1) Apply preheating for 60 to 120 seconds at a temperature of 150°C.

(2) For lead insertion-type packages, mount the device within 10 seconds of solder flow with a maximum temperature of 260°C at the stopper or at a location more than 1.5mm from the body.

(3) For surface-mount packages, mount the device within 5 seconds at a temperature of 250°C or less in order to avoid thermal stress.

(4) Figure 3.5 shows an example of the temperature profile of solder flow for a surface-mount package.

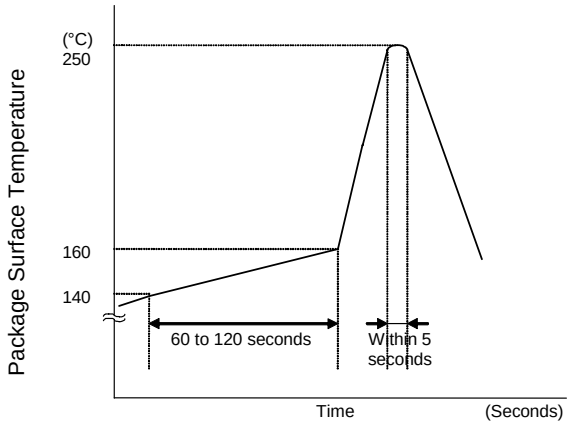


Figure 3.5 Example of Surface-Mount Package Temperature Profile

This profile is based on the device's maximum heat resistance guaranteed value.

Set the preheat temperature/heating temperature to the optimum temperature corresponding to the solder paste type used by the customer within the above-described profile.

3.5.4 Flux Cleaning

(1) When cleaning circuit boards to remove flux, make sure that no reactive ions such as sodium or chlorine remain. Some organic solvents react with water to generate hydrogen chloride and other corrosive gases which can result in device deterioration.

(2) When washing devices with water, make sure that no reactive ions such as sodium or chlorine remain particularly.

(3) When washing devices, do not rub markings with a brush or with your hand while the cleansing liquid is still on the device. Doing so can rub off the markings.

(4) Dip cleaning, shower cleaning and steam cleaning processes are performed based on the chemical action of a solvent. When immersing devices in a solvent or steam bath, complete the cleaning for a period of one minute or less at a liquid temperature of 50°C or less, taking into consideration the effects on the devices.

(5) Avoid use of ultrasonic cleaning with hermetically sealed ceramic packages such as a leadless chip carrier (LCC), pin grid array (PGA) or charge-coupled device (CCD). Using the ultrasonic cleaning may cause the internal wires to become disconnected due to resonance. Even if a device package allows ultrasonic cleaning, keep the duration of ultrasonic cleaning in a brief time. Long hours of ultrasonic cleaning may deteriorate the adhesion between the mold resin and frame material.

The basic recommended conditions are as follows:

Recommended Ultrasonic Cleaning Conditions

Frequency: 27 to 29kHz

Ultrasonic output: 15W/L or less

Cleaning time: 30 seconds or less

Suspend the printed circuit board in the solvent bath to ensure that the circuit board and device do not come in direct contact with the ultrasonic vibrator.

3.5.5 No Cleaning

It is recommended that you clean analog devices and high-speed devices. If such devices are not cleaned, flux may cause minute leakage between leads or migration, depending on the flux grade. Be sure therefore to check cleanliness at the time of use. If you are considering no cleaning, be sure to use a flux that does not require cleaning.

3.5.6 Tape Carrier Packages (TCPs) Mounting

(1) When tape carrier packages are mounted, measures must be taken to prevent electrostatic breakdown of the devices.

(2) When separating devices from tape, or carrying out outer lead bonding (OLB) mounting, be sure to take work safety into consideration.

(3) The base film, which is made of polyimide, is hard and thin. Be careful not to injury yourself or damage any objects during handling.

(4) When punching tape, take countermeasures to prevent minute broken pieces from scattering. Scattered pieces may cause injury.

(5) Appropriately treat the tape, reels and spacers left after separating the device as industrial waste.

(6) With tape carrier package (TCPs) devices, the backside of the LSI chips is exposed. To ensure that the chip will not crack, mount the device so that mechanical shock is not applied to the LSI backside. In addition, electrical contact may also cause LSI failure. Mount the device so that there is no electrical contact with the backside of the LSI chip.

If you are mounting the backside of the LSI chip to improve device characteristics, please contact your nearest Toshiba office or distributor in advance.

3.5.7 Chips Mounting

Devices delivered in chip form readily deteriorate or become damaged due to external factors in comparison with plastic-packaged products. Attention is therefore required during handling.

(1) Mount devices in a properly maintained environment so that the chip will not be exposed to contaminated ambient air or other substances.

(2) When handling chips, be careful not to expose the chips to static electricity. In particular, measures must be taken to prevent electrostatic breakdown during chip mounting. For this purpose, it is recommended that you mount all peripheral devices before you mount the chips.

(3) Use chip mounting circuit boards (such as PCBs) that do not have any chemical residue on them (such as the chemicals used during PCB etching).

(4) When mounting chips, use the method of assembly that is most suitable for achieving the appropriate electrical, thermal and mechanical characteristics of the semiconductor product used.

* For chip details, refer to the relevant specification sheet.

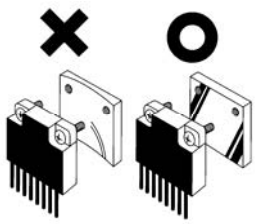
3.5.8 Circuit Board Coating

When using devices that require high reliability or devices used under extreme environments (where moisture, corrosive gas or dust is present), circuit boards are sometimes coated with a moisture-proof coating. When using a coating resin, choose the coating resin which results in minimal stress to the device.

3.5.9 Heat Sinks

(1) When installing a heat sink to a device, use the specified accessories. In addition, be careful not to apply excessive force to the device during installation.

(2) When installing a device to a heat sink by fixing it in two or more locations, do not tighten one location to the specified torque while the rest are left not tightened. Rather, lightly tighten all locations evenly first and tighten all locations to the specified torque by rotation.



(3) Drill screw holes in the heat since as specified, and smooth the surface of the device installation area by removing burrs and protrusions or indentations.

(4) Thinly applying silicone grease between the heat sink makes device better to improve heat conductivity compared with no grease. If you choose to apply the silicone grease, use a non-volatile type. Volatile type silicone grease can cause

cracks over time, resulting in the deterioration of the heat radiation effect.

(5) With plastic-packaged devices, the base oil of some silicone grease compounds penetrates the package interior, significantly reducing the lifetime of the device. We ask therefore that you use the recommended silicon grease YG6260 from GE Toshiba Silicone. If you choose to use another product, select one that is equivalent to the Toshiba Silicone product.

(6) During device operation, heat sinks become very hot. Be careful not to touch them. A burn may result.

3.5.10 Tightening Torque

(1) Tighten screws to a tightening torque that is within the specified values described in the individual technical datasheets and databooks for the device used.

(2) Be careful not to allow a pneumatic screwdriver to come in contact with devices. Device damage may result.

3.5.11 Repeated Device Mounting and Usage

Do not remount or reuse devices that have histories such as that described below. These devices may cause significant problems with regard to device characteristics and reliability.

(1) Devices that have been removed from the board after soldering.

(2) Devices that have been inserted in the wrong orientation or with reverse polarity and charged.

(3) Devices that have undergone lead forming more than once.

3.6 Operating Environment

3.6.1 Temperature

Semiconductor devices are generally more sensitive to temperature than other electromechanical parts. The various electrical characteristics of a semiconductor device are restricted by the operating temperature. It is therefore necessary to understand the temperature characteristics of a device and incorporate derating into the device design in advance. When a device is used at a temperature outside the specified operating range, electrical characteristics will not be realized and device deterioration will occur more rapidly.

3.6.2 Humidity

Plastic package devices are sometimes not completely sealed.

When these devices are used for an extended period of time under high humidity, moisture can seep into the device and cause semiconductor chip deterioration or failure. Furthermore, when devices are mounted on a regular printed circuit board, the impedance between wiring can decrease under high humidity. In systems with a high signal-source impedance, circuit board leakage or leakage between device leads can cause malfunction. In such a case, moisture-proof treatment to the device surface should be considered. On the other hand, operation under low humidity can damage a device due to the occurrence of electrostatic discharge. Unless moisture-proof treatments have been specifically taken, use devices within the humidity range of 40 to 60%.

3.6.3 Corrosive Gases

Devices react to corrosive gases may cause deteriorating device characteristics. For example, consideration must be given to lead corrosion and leakage between leads caused by the chemical reaction that occurs when a device is placed near a rubber product. The reason is that the rubber product will not only produce condensation but also generate sulfur-bearing corrosive gases under high-humidity conditions.

3.6.4 Radioactive and Cosmic Rays

Standard devices are not designed with protection against radioactive and cosmic rays. Devices must therefore be shielded if the device will be used in environments that may result in exposure to radioactive or cosmic rays above the levels that exist in the natural environment.

3.6.5 Strong Electrical and Magnetic Fields

Devices exposed to magnetic fields can undergo a polarization phenomenon in the plastic material or within the IC chip, which gives rise to abnormal conditions such as impedance changes or leak current increases. Malfunctions have been reported in LSIs mounted near television deflection yokes. In such cases, the device installation location must be changed or the device must be shielded against the electrical or magnetic field. Shielding against magnetism is especially required in an alternating magnetic field due to the electromotive forces generated.

3.6.6 Interference from Light (such as Ultraviolet Rays, Sunlight, Fluorescent Lamps, Incandescent Lamps)

Light striking a semiconductor device generates electromotive force due to photoelectric effects, sometimes causing malfunction. Devices in which the chip is visible through the package are especially affected by such light. When designing the circuits, make sure that the devices are protected against light interference. Not just optical semiconductor devices, but all types of devices are affected by light.

3.6.7 Dust and Oil

Similar to corrosive gases, dust and oil cause chemical reactions in semiconductor products, sometimes adversely affecting product characteristics. Be sure to use semiconductor products in an environment that will not result in dust or oil adhesion. Solvent and oil contained in heat release sheets similarly may result in semiconductor product quality deterioration, characteristic deterioration or disconnection. Be sure to use such products with care.

3.6.8 Smoke and Ignition

Semiconductor devices and modularized devices are not noncombustible; they can emit smoke or ignite when excessive current or failure occurs. When this happens, poisonous gases may be produced.

Be sure to develop a safe design that protects the device from excessive current so as to ensure excessive current does not flow within the device during operation or at the time of failure.

To prevent the propagation of fire caused by a smoking or ignited Toshiba product and to ensure that Toshiba products do not emit smoke or ignite due to surrounding conditions, do not use Toshiba products in close proximity to combustible thing, heat-generating thing, igniting materials or flammable materials.

3.7 Disposal

Each country and region has laws and regulations for the proper disposal of devices and packing materials. Be sure to follow these laws and regulations at the time of disposal.

4. Precautions and Usage Considerations Specific to Each Product Group

This section describes the matters specific to each product group which need to be taken into consideration. The precautions described in this section take precedence over those described in Section 3, “General Safety Precautions and Usage Considerations.”

4.1 Microcomputers

4.1.1 Design

(1) Use of Crystal Oscillators Other than Those Recommended

For components such as crystal oscillators that are used in the oscillation circuit of microcomputer products, use the components with usage conditions described in the individual technical datasheets and databooks. If you plan on using components with usage conditions other than those recommended, contact the our engineering department stated in the individual technical datasheet or databook, or consult with the oscillator manufacturer.

(2) Undefined Functions

Microcomputer products have commands that are not individually defined for each product (i.e., undefined commands). These products also similarly have undefined functions (for instance, bits to which functions are not assigned in the register).

Refer to the product’s individual technical datasheets and databooks and do not use the undefined commands and undefined functions.

TC86C001FG

2006-10

Rev. 2.1

1. General Description

The TC86C001FG (GOKU-S) is a highly integrated and highly performance solution. It provides most popular and widely used 5 interfaces – PCI Interface, ATA/ATAPI Host Controller, USB Host Controller, USB Device Controller, I^2C bus or Serial I/O Interface.

The GOKU-S provides the best solution for a wide range of digital products such as set-top boxes, personal video recorder, information appliances, Multifunction Printer.

2. Outline and Features

GOKU-S is a Multi-function PCI device that consists of ATA/ATAPI Function, USB Host Function, USB Device Function, and I^2C bus Bus / Serial I/O / General Purpose I/O Function.

- ATA/ATAPI Host Controller

  Single channel for up to two devices (Master/Slave)

  Ultra DMA mode4 (ATA-5) (up to 66.67 Mbyte/sec)

  Support Ultra DMA Mode 0, 1, 2, 3, 4

  PIO Mode 0 to 4, Multiword DMA Mode 0, 1 and 2

  Compliant with PCI IDE Controller Specification Rev1.0 (Native-PCI only)

  Compliant Programming Interface for Bus Master IDE Controller Rev1.0

  DMA controller with 512 byte x 2 FIFOs

  ATAPI Overlap command not support

- USB Host Controller

  Compliant with Universal Serial Bus Specification Rev1.1

  Compliant with OpenHCI for USB Release 1.0a

  2 ports for serial transfers at 12 Mbit/sec (Full Speed) or 1.5 Mbit/sec (Low Speed)

  Over current protection and Power control for each port

- USB Device Controller

  Compliant with Universal Serial Bus Specification Rev1.1

  1 port for serial transfers at 12 Mbit/sec (Full Speed) only.

  Supports three types of transfer, Control transfer, Bulk transfer, and Interrupt transfer

  Supports one control endpoint (EP0), three endpoint (EP1, EP2, EP3) for selectable Bulk/Interrupt transfer

  EP0: Control transfer (64 byte FIFO)

  EP1, 2: Bulk/Interrupt transfer (Receive/Transmit selectable, 64 byte + 64 byte FIFOs)

  EP3: Bulk/Interrupt transfer (64 byte FIFO)

  Isochronous transfers not support

- I^2C, Serial I/O and GPIO Controller (Multiplexed I^2C, Serial I/O, GPIO)

  I^2C (Single Master/Slave) 1 ch

  Serial I/O (Full duplex UART) 1 ch

  GPIO 5 pin (Max)

  Independent selection of direction of pins and output port type (totem-pole or open-drain output) on a per bit basis.

- PCI Interface

  32-bit PCI bus interface at 33 MHz PCI operation., 3.3 V Interface

  Fully compliant with PCI Local Bus Specification Rev. 2.2

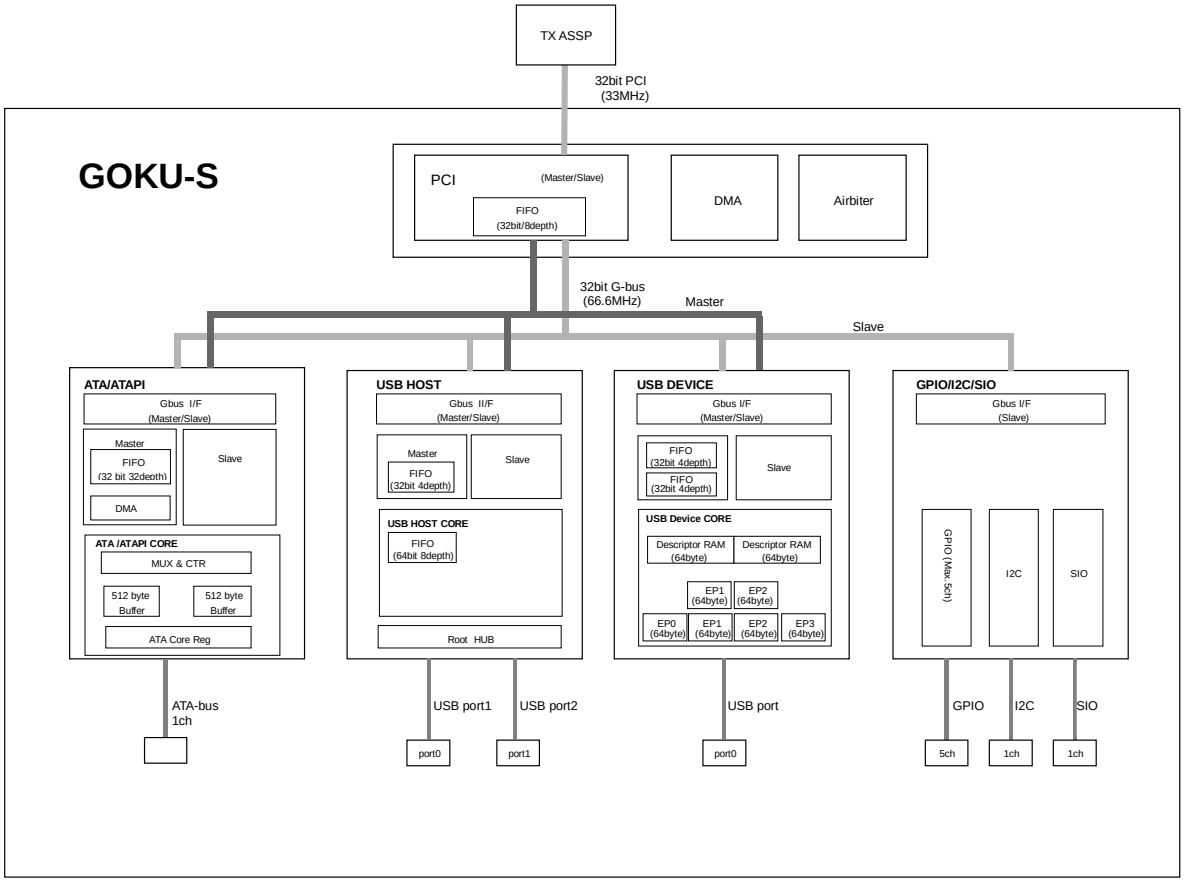
  Fully compliant with PCI Bus Power Management interface Specification Rev. 1.1

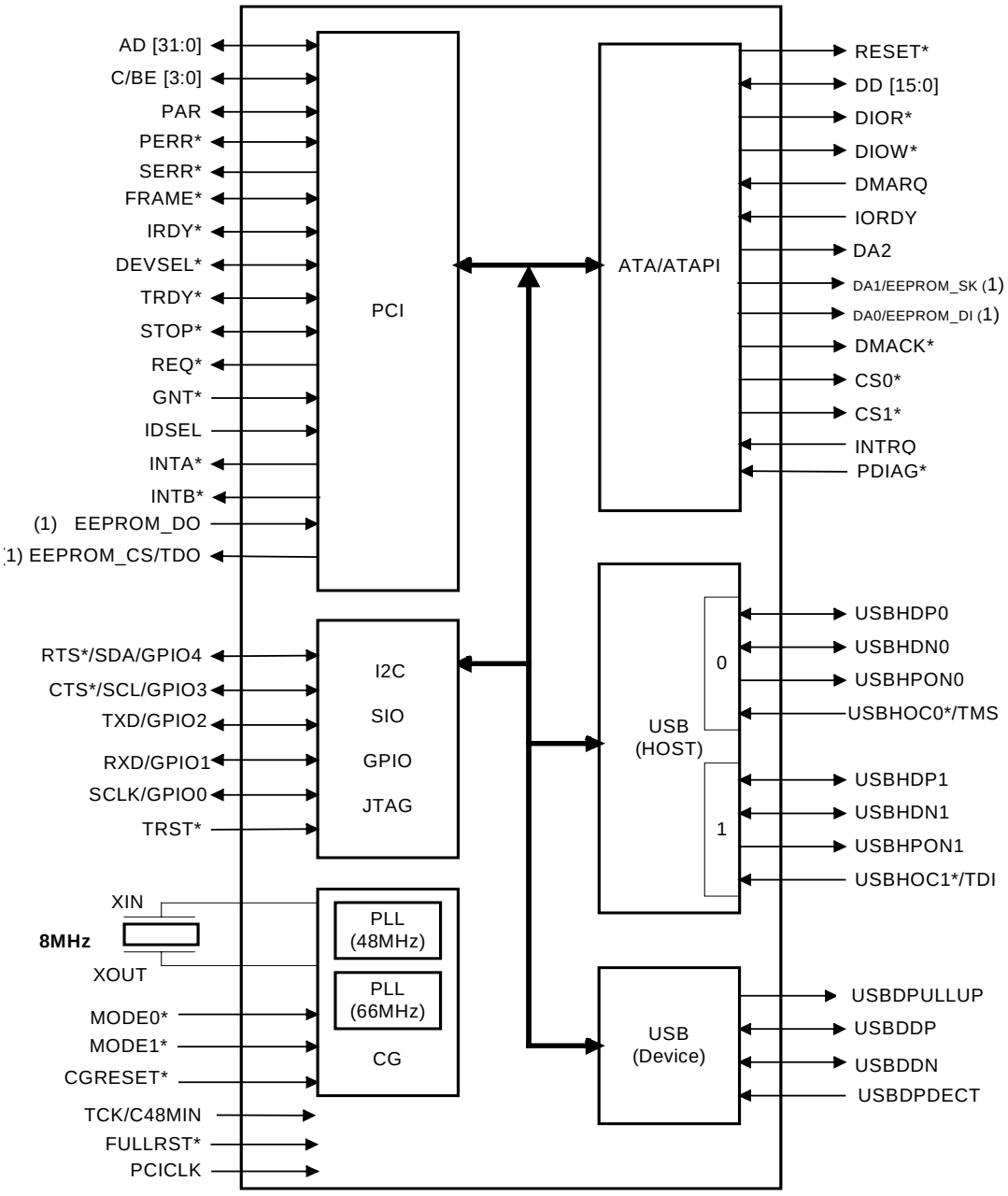
  Loadable PCI Configuration Space with EEPROM Interface

  Supports 2 Interrupts function (INTA and INTB)

  PCI Arbiter with Round-robin priority arbitration for the internal PCI devices

3. Structure


TX ASSP
32bit PCI
(33MHz)
GOKU-S
PCI
(Master/Slave)
FIFO
(32bit/8depth)
DMA
Airbiter
32bit G-bus
(66.6MHz)
Master
Slave
ATA/ATAPI
Gbus I/F
(Master/Slave)
Master
FIFO
(32 bit 32depth)
DMA
Slave
ATA /ATAPI CORE
MUX & CTR
512 byte
Buffer
512 byte
Buffer
ATA Core Reg
ATA-bus
1ch
USB HOST
Gbus II/F
(Master/Slave)
Master
FIFO
(32bit 4depth)
Slave
USB HOST CORE
FIFO
(64bit 8depth)
Root HUB
port0
port1
USB port1
USB port2
USB DEVICE
Gbus I/F
(Master/Slave)
FIFO
(32bit 4depth)
FIFO
(32bit 4depth)
Slave
USB Device CORE
Descriptor RAM
(64byte)
Descriptor RAM
(64byte)
EP1
(64byte)
EP2
(64byte)
EP0
(64byte)
EP1
(64byte)
EP2
(64byte)
EP3
(64byte)
port0
USB port
GPIO/I2C/SIO
Gbus I/F
(Slave)
GPIO (Max. 5ch)
I2C
SIO
GPIO
I2C
SIO
5ch
1ch
1ch



(1) EEPROM interface Signals (PCI INTERFACE)

4. Pins

Note: In the following table, “PU” in the type column indicates the presence of internal pull-up resistance, and “OD” indicates open drain. The asterisk “*” at the end of a signal name indicates that that signal is an Active Low signal.

4.1 PCI Interface

Signal Name	Type	Description
PCICLK	Input	PCI CLOCK (33 MHz) PCI bus clock signals
AD[31:0]	Input/Output	PCI ADDRESS/DATA Multiplexed address and data bus
C/BE[3:0]	Input/Output	BUS COMMAND AND BYTE ENABLES Command and byte enable signals
PAR	Input/Output	CALCULATED PARITY SIGNAL
PERR*	Input/Output	PARITY ERROR
SERR*	Output (OD)	SYSTEM ERROR
FRAME*	Input/Output	CYCLE FRAME
IRDY*	Input/Output	INITIATOR READY
DEVSEL*	Input/Output	DEVICE SELECT
TRDY*	Input/Output	TARGET READY
STOP*	Input/Output	STOP
REQ*	Output	BUS REQUEST
GNT*	Input	BUS GRANT
IDSEL	Input	INITIALIZATION DEVICE SELECT
INTA*	Output (OD)	INTERRUPT A
INTB*	Output (OD)	INTERRUPT B
EEPROM_DO	Input (PU)	EEPROM Data from the EEPROM to the core When using JTAG: Set to “Low” Normal Mode and when not using EEPROM: Set to “High (pull up)”
EEPROM_CS	Output	EEPROM Chip Select

4.2 ATA/ATAPI Interface (Multiplexed with Serial EEPROM Interface)

Signal Name	Type	Description
RESET*	Output	RESET
DD[15:0]	Input/Output	DATA
DIOR*	Output	IO READ
DIOW*	Output	IO WRITE
DMARQ	Input	DMA REQUEST
IORDY	Input	IO CHANNEL READY
DA2	Output	ADDRESS2
DA1 /EEPROM_SK	Output	ADDRESS1 /EEPROM_CLOCK
DA0 /EEPROM_DI	Output	ADDRESS0 /EEPROM Data to the EEPROM from the core
DMACK*	Output	DMA ACKNOWLEDGE
CS0*	Output	CHIP SELECT
CS1*	Output	CHIP SELECT
INTRQ	Input	INTERRUPT REQUEST
PDIAG*	Input	PASS DIAGNOSTIC

4.3 USB (Host) Interface

Signal Name	Type	Description
USBHDP0	Input/Output	USB HOST PORT 0 DATA +
USBHDN0	Input/Output	USB HOST PORT 0 DATA -
USBHPON0	Output	POWER ON CONTROL (PORT 0)
USBHOC0*	Input (PU)	OVER CURRENT DETECT (PORT 0)
USBHDP1	Input/Output	USB HOST PORT 1 DATA +
USBHDN1	Input/Output	USB HOST PORT 1 DATA -
USBHPON1	Output	POWER ON CONTROL (PORT 1)
USBHOC1*	Input (PU)	OVER CURRENT DETECT (PORT 1)

4.4 USB (Device) Interface

Signal Name	Type	Description
USBDDP	Input/Output	USB DEVICE PORT DATA +
USBDDN	Input/Output	USB DEVICE PORT DATA -
USBDPULLUP	Output	USB DEVICE PULL UP
USBDPDECT	Input	USB DEVICE POWER DETECT

4.5 I2C/SIO/GPIO Interface

Signal Name	Type	Description
GPIO4 (Mode0, 2)	Input (PU)/Output (OD)	GENERAL PURPOSE INPUT/OUTPUT (programmable open drain by GPIOOD Register)
/SDA (Mode1, 3, 4)	Input (PU)/Output (OD)	/SERIAL DATA LINE
/RTS* (Mode5, 6)	Output	/SIO REQUEST TO SEND
GPIO3 (Mode0, 2)	Input (PU)/Output (OD)	GENERAL PURPOSE INPUT/OUTPUT OUTPUT (programmable open drain by GPIOOD Register)
/SCL (Mode1, 3, 4)	Input (PU)/Output (OD)	/SERIAL CLOCK LINE
/CTS* (Mode5, 6)	Input (PU)	/SIO CLEAR TO SEND
GPIO2 (Mode0, 1)	Input (PU)/Output (OD)	GENERAL PURPOSE INPUT/OUTPUT OUTPUT (programmable open drain by GPIOOD Register)
/TXD (Mode 2-6)	Output	/SIO TRANSMIT DATA
GPIO1 (Mode0, 1)	Input (PU)/Output (OD)	GENERAL PURPOSE INPUT/OUTPUT OUTPUT (programmable open drain by GPIOOD Register)
/RXD (Mode 2-6)	Input (PU)	/SIO RECEIVE DATA
GPIO0 (Mode0-3, 5)	Input (PU)/Output (OD)	GENERAL PURPOSE INPUT/OUTPUT OUTPUT (programmable open drain by GPIOOD Register)
/SCLK (Mode 4, 6)	Input (PU)	/EXTERNAL SERIAL CLOCK

4.6 JTAG Interface (Multiplexed with Other Signals)

Signal Name	Type	Description
TRST*/–	Input (PU)	TEST RESET INPUT
TDO/EEPROM_CS	Output	JTAG DATA OUTPUT
TDI/USBHOC1*	Input	JTAG DATA INPUT
TMS/USBHOC0*	Input	JTAG COMMAND
TCK/C48MIN (For Test use)	Input (PU)	JTAG CLOCK INPUT/USB internal CLK 48 MHz (For test use)

4.7 Clock

Signal Name	Type	Description
XIN	Input	Crystal Input (8 MHz)
XOUT	Output	Crystal Output (8 MHz)
CGRESET*	Input (PU)	CG RESET

4.8 Reset

Signal Name	Type	Description
FULLRST*	Input (PU)	FULL RESET

4.9 Test

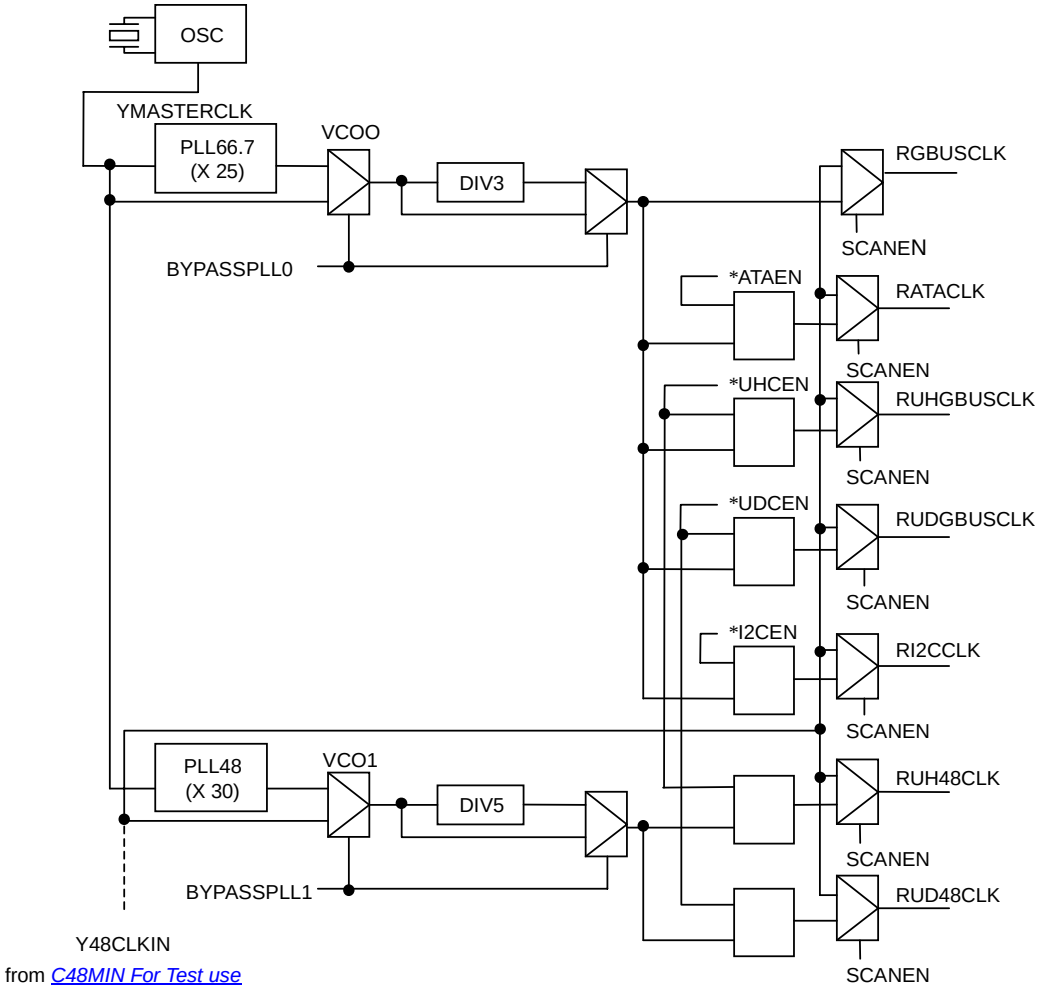
Signal Name	Type	Description
MODE0*	Input (PU)	MODE SELECT Normal Mode: Set this pin to “High” JTAG Mode: Set this pin to “Low”
MODE1*	Input (PU)	MODE SELECT Normal Mode: Set this pin to “High” JTAG Mode: Set this pin to “Low”

4.10 Power/Ground

Signal Name	Type	Description
V_{CCINT}	Input	VCC VOLTAGE PINS FOR INTERNAL CORE (1.5 V)
V_{CCIO}	Input	VCC VOLTAGE PINS FOR I/O SIGNALS (3.3 V)
V_{SS}	Input	GROUND PINS
PLL1VDD_A	Input	VDD VOLTAGE PIN FOR PLL1 (1.5 V)
PLL1VSS_A	Input	GROUND PIN FOR PLL1
PLL2VDD_A	Input	VDD VOLTAGE PIN FOR PLL2 (1.5 V)
PLL2VSS_A	Input	GROUND PIN FOR PLL2

5. Clocks

5.1 Goku-S Clock Signals



6. Bus Master IDE Controller (Function 0)

6.1 Functional Description

6.1.1 Initialization

This section explains the procedure for initializing this controller.

6.1.1.1 Reset Sequence

Apply a reset to the ATA/ATAPI I/F Controller block and the HDD in the following sequence. The PCI DMA Controller block is not reset.

(1) Reset the ATA/ATAPI I/F Controller block.
Set Soft Reset (bit[15]) of the System Control 1 Register (+0x00). The two following registers are initialized to 0x0100.

<System Control I/O Register Space>

Transfer Word Count 1 Register (+0x04)

Transfer Word Count 2 Register (+0x06)

(2) Waiting time is unnecessary.

(3) Reset the FIFO.
Set FIFO Reset (bit[14]) of the System Control 1 Register (+0x00).

(4) Waiting time is unnecessary.

(5) Perform hardware reset on the device.
Set ATA Hard Reset (bit[11]) of the System Control 1 Register (+0x00).

(6) Wait for 25 μs or more

(Note: This value “25 μs or more” is provided by the standard “AT Attachment with Packet Interface-5”. However, there are some devices that a start after reset takes time. Please adjust time in accordance with devices that you connect.)

(7) Cancel all resets.
Write 0x0000 to the System Control 1 Register (+0x00).

6.1.1.2 Identifying the Cable Type

You can monitor the CBLIDE signal from PDIAGN (bit[13]) of the System Control 1 Register (+0x00).

0: 80-conductor cable (You can use UltraDMA Mode3,4)

1: 40-conductor cable (You cannot use UltraDMA Mode3,4)

6.1.1.3 Setting the Transfer Mode

This section explains the Command transfer modes and data transfer modes that this controller uses.

Set the Master Device transfer mode in the System Control 1 Register (+0x00) and the Slave Device transfer mode in the System Control 2 Register (+0x02). When you select the device (Master or Slave), the transfer mode of that device becomes valid.

You are recommended to set the PIO transfer mode to the slowest of the Master device or the Slave device, and use the same transfer mode for both. You can set the Master Device and the Slave Device to different transfer modes.

(1) Setting the Transfer Mode of the Master Device

This setting becomes valid when you select a Master device.

- PIO Transfer Mode Setup Register

  System Control 1 Register (+0x00) -- PIO Transfer Mode Select Master (bit[10:8])

  0x00: PIO Mode 0
  0x01: PIO Mode 1
  0x02: PIO Mode 2
  0x03: PIO Mode 3
  0x04: PIO Mode 4

- DMA Transfer Mode Setup Register

  System Control 1 Register (+0x00) -- DMA Transfer Mode Select Master (bit[7:4])

  0x00: Reserved
  0x01: Reserved
  0x02: Reserved
  0x03: Reserved
  0x04: Reserved
  0x05: Multi Word DMA Mode 0
  0x06: Multi Word DMA Mode 1
  0x07: Multi Word DMA Mode 2
  0x08: Ultra DMA Mode 0
  0x09: Ultra DMA Mode 1
  0x0A: Ultra DMA Mode 2
  0x0B: Ultra DMA Mode 3
  0x0C: Ultra DMA Mode 4

(2) Setting the Slave Device transfer mode

This setting becomes valid when you select a Slave device.

- PIO Transfer Mode Setup Register

  System Control2 Register (+0x02) -- PIO Transfer Mode Select Slave (bit[10:8])
  Selection choices are similar to PIO Transfer Mode Select Master (bit[10:8]).

- DMA Transfer Mode Setup Register

  System Control2 Register (+0x02) -- DMA Transfer Mode Select Slave (bit[7:4])
  Selection choices are similar to DMA Transfer Mode Select Master (bit[7:4]).

6.1.1.4 Setting the Transfer Word Count

This section explains the procedure for setting the transfer word count that this controller uses during DMA transfer.

Set the word count (16 bits) for each device sector (block) in the transfer word count. Set the Master Device transfer word count in the Transfer Word Count 1 Register (+0x04) and the Slave Device transfer word count in the Transfer Word Count 2 Register (+0x06). This setting is loaded during DMA Data transfer, in other words, when the START_STOPBM bit (bit[0]) is set.

(1) Setting the Master Device Transfer Word Count

Set the word count for each sector (block) in the Master Device as the transfer word count.

- Transfer Word Count Setup Register

  Transfer Word Count 1 Register (+0x04)

(2) Setting the Slave Device Transfer Word Count

Set the word count for each sector (block) in the Slave Device as the transfer word count.

- Transfer Word Count Setup Register

  Transfer Word Count 2 Register (+0x06)

6.1.1.5 Setting Sector Count Control

This section explains the procedure for setting sector count control that this controller uses during DMA transfer. Select the method of loading the sector count value of the DMA transfer command in this controller. There are two methods of doing this, so it is possible to have separate settings for a Master Device and a Slave device.

One method is for this controller to automatically load the value of the ATA Sector Count Register (+0x02). Select this method for ATA devices.

The other method is for this controller to load the value of the Sector Count Register (+0x0A) of the System Control I/O Space. The Host must set the sector count transferred to the Sector Count Register (+0x0A) in advance. Select this method for ATAPI devices.

(1) Setting the Sector Count control of the Master Device

Set Sector Count Control of the Master Device to Sector Count Control Register (+0x0C) – Sector Count Control Master (bit[0]).

0: Load the value of the ATA Sector Count Register.

1: Load the value of the Sector Count Register of this controller.

(2) Setting the Sector Count control of the Slave Device

Set Sector Count Control of the Slave Device to Sector Count Control Register (+0x0C) – Sector Count Control Slave (bit[1])

6.1.1.6 Big Drive (Over 137 GB HDD)

When using new DMA command of Big Drive.

(1) At the time of initialization, the Sector Count control Field of the Sector Count control Register (SYS_CNT_IO_BASE+0x0C) is set to 1.

(2) Even before a DMA transfer start, the Sector Count Register (SYS_CNT_IO_BASE+0x0A) is set to the same value as the sector count value set as the ATA sector count register.

At the time of new PIO command of big Drive use, an additional register setup is unnecessary.

6.1.2 PIO Data Transfer

This section explains the settings this controller requires when performing PIO data transfer.

6.1.2.1 Setting the Transfer Mode

Refer to 6.1.1.3 Setting the Transfer Mode for more information regarding setting the transfer mode.

6.1.2.2 ATAPI Packet Write Function

When sending an ATAPI packet command, you can use the ATAPI Packet Write Register (+0x08) instead of the ATA Command Register (+0x07).

6.1.2.3 Program Sequence Example

(1) Clear INT_IDE (Interrupt) (bit[2]) of the Status Register (+0x02).

(2) Issue a PIO command.

(3) Perform PIO transfer.

(4) An interrupt (INTA* or INTB*) is issued when transfer ends.

(5) The Interrupt Handler uses INT_IDE (bit[2]) of the Status Register (+0x02) to judge if this controller issued an interrupt.

(6) Read the ATA Status Register (+0x07) to judge whether the transfer ended normally.

6.1.3 DMA Data Transfer

This section explains the settings this controller requires when performing DMA data transfer.

6.1.3.1 Setting the Transfer Mode

Refer to 6.1.1.3 Setting the Transfer Mode for more information regarding setting the transfer mode.

6.1.3.2 Setting the Sector Count

This section explains the setting of the sector count value this controller uses when performing DMA data transfer.

You must set in advance the number of sectors transferred to the Sector Count Register (+0x0A) if the sector count control setting selects the method of loading the value of the Sector Count Register (+0x0A) of the System Control I/O Space. This setting is loaded during DMA data transfer, in other words, when the START_STOPBM bit (bit[0]) is set.

Refer to 6.1.1.5 Setting Sector Count Control for more information regarding sector count control.

6.1.3.3 ATAPI Packet Write Function

When sending an ATAPI packet command, you can use the ATAPI Packet Write Register (+0x08) instead of the ATA Command Register (+0x07).

6.1.3.4 Note about the DMA data transfer of ATAPI device

This controller doesn't support the DMA data transfer of the size which doesn't arrive at one block (sector). Don't use the DMA data transfer, but use the PIO data transfer when you execute packet command accompanied with the data transfer of the size which doesn't arrive at one block.

(In ATA(IDE) device, there is not packet command accompanied with the data transfer of the size which doesn't arrive at one block. It doesn't need to take this item into consideration.)

(1) Packet command accompanied with the data transfer of the size which doesn't arrive at one block
(The command which doesn’t specify transfer length in the block unit applies.)

-MODE SENSE Command (0x55)
-INQURIY Command (0x12)
-REQUEST SENSE Command (0x03)
-READ CAPACITY Command (0x25)
etc.

(2) Packet command accompanied with the data transfer of the size which is one block. (The command which doesn't specify transfer length in the block unit applies.)

-READ(10) Command (0x28)
-READ(12) Command (0x0A8)
-WRITE(10) Command (0x2A)
-WRITE(12) Command (0x0AA)
etc.

6.1.3.5 Programming Sequence Example

(1) Set up the registers of this controller.
System Control 2 Register (+0x02)
When an ATAPI device sets the sector count value:
Sector Count Register (+0x0A)

(2) Create a PRD table in system memory. This table consists of several Physical Region Descriptors (PRD) that memory regions describe during data transfer. This table must exist in double-word-alignment. Also, this table must not exceed 64 K boundaries.

Physical Region Descriptor Table (PRD × *n*)

+0x00	Physical Region Descriptor (PRD) 1
+0x08	Physical Region Descriptor (PRD) 2
+(n-2)*0x08	Physical Region Descriptor (PRD) n-1
+(n-1)*0x08	Physical Region Descriptor (PRD) n

(3) Set the start address (32-bit) of the PRD table in the Descriptor Table Pointer Register (+0x04).

(4) Set R_OR_WCTR (Read or Write Control) (bit[3]) of the Command Register (+0x00).

(5) Clear INT_IDE (Interrupt) (bit[2]) and ERROR (bit[1]) of the Status Register (+0x02).

(6) Issue a DMA command.

(7) Set START_STOPBM (Start/Stop Bus Master) (bit[0]) of the Command Register (+0x00).

(8) This controller performs DMA transfer.

(9) An interrupt (INTA* or INTB*) is issued when transfer ends.

(10) The Interrupt Handler uses INT_IDE (bit[2]) of the Status Register (+0x02) to judge whether this controller has issued an interrupt.

(11) Reset START_STOPBM (Start/Stop Bus Master) (bit[0]) of the Command Register (+0x00).

(12) Read the Status Register (+0x02) and the ATA Status Register (+0x07) to judge whether transfer ended normally.

6.1.4 Interrupts

This section explains interrupts that are related to this controller.

6.1.4.1 Selecting PCI Interrupt Pins

You can use the following register to select the PCI interrupt pins that this controller uses.

PCI Interrupt Pin Selectable Register (0x44)

0x01: INTA*

0x02: INTB*

6.1.4.2 Interrupt Status

When performing read transfer from an IDE device, the IDE device issues an interrupt (IDE_IRQ), the data in the FIFO inside this controller is written to memory, then this controller issues an interrupt (INTA*, INTB*).

Bus Master IDE Status Register (0x02) INT_IDE bit[2]

1: IDE_IRQ is asserted

6.1.4.3 Clearing Interrupts

Clearing an interrupt (IDE_IRQ) from an IDE device clears interrupts (INTA*, INTB*) from this controller. Interrupts from this controller are not cleared even if interrupt status INT_IDE (bit[2]) is cleared.

6.2 Register Description

The Bus Master IDE Controller has one PCI configuration space, space for command I/O registers, space for control I/O registers, space for bus master I/O registers and space for system control I/O registers. The I/O registers are accessed after setting the base address in the configuration register.

This module is compliant to PCI IDE Controller Specification Rev1.0 and Programming Interface for Bus Master IDE Controller Rev.1.0. This module provides an industrial standardized Scatter/Gather host DMA mechanism that complies with Revision 1.0 of the Programming Interfaces for Bus Master IDE Controller. This module has 512 × 2 byte FIFO.

Table 6.2.1 Register Access Type

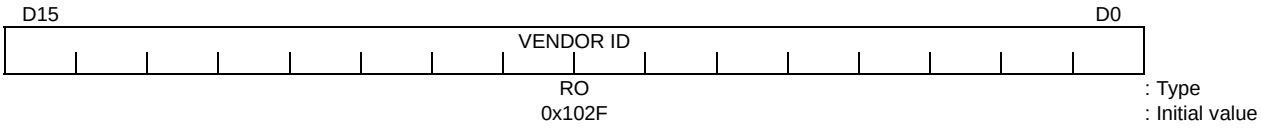
Access Type	Meaning
RO	Read-only
WO	Write-only
RW	Can both be read and written to.
RWC	Can both be read and written to. Cleared when written to.

6.2.1 Configuration Map

23	15	07	00	
Device ID		Vendor ID		0x00
Status		Command		0x04
Class code			Revision ID	0x08
BIST	Header type	Latency timer	Cache Line Size	0x0C
Command IO base address				0x10
Control IO base address				0x14
				0x18
				0x1C
Bus Master IDE IO base address				0x20
System Control IO base address				0x24
				0x28
Sub System ID		Sub Vendor ID		0x2C
				0x30
			Capability Pointer	0x34
				0x38
Maximum Latency	Minimum GNT	PCI Interrupt pin	PCI Interrupt line	0x3C
		Retry Timeout	TRDY Timeout	0x40
			PCI Interrupt pin select	0x44
				0x48-0xD8
Power Management Capabilities		Next Item Pointer	Capability ID	0xDC
Data	PMCSR BSE	Power Management Control/Status		0xE0
				0xE4-0xFC

6.2.2 Configuration Register Details

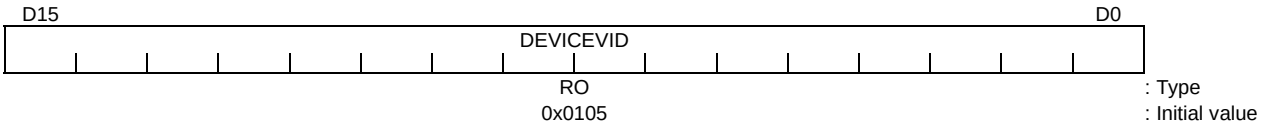
6.2.2.1 Vendor ID Register (0x00 – 0x01)



Bits	Mnemonic	Field Name	Description
D15:D0	VENDOR ID	Vendor ID	Vendor ID This value 0x102F represents the Toshiba Semiconductor Company.

Figure 6.2.1 Vendor ID Register

6.2.2.2 Device ID Register (0x02 – 0x03)



Bits	Mnemonic	Field Name	Description
D15:D0	DEVICE ID	Device ID	Device ID This value 0x0105 represents the Bus Master IDE Controller.

Figure 6.2.2 Device ID Register

6.2.2.3 Command Register (0x04 – 0x05)

D15	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	
Reserved		FBBE	SER	WCC	PER	VGA	MW1	SP	BM	MS	IOS	
RO		RW	RW	RO	RW	RO	RW	RO	RW	RO	RW	: Type
0x00		0	0	0	0	0	0	0	0	0	0	: Initial value

Bits	Mnemonic	Field Name	Description
D15:D10		Reserved	
D9	FBBE	Fast Back-to-Back Enable	Fast Back-to-Back Enable This bit controls whether or not a master can do fast back-to-back transactions to different devices. A value of 1 means the master is allowed to generate fast back-to-back transactions to different agents. A value of 0 means fast back-to-back transactions are only allowed to the same agent.
D8	SER	SERR Enable	SERR Enable This is the enable bit for the SERR* driver. All devices that have an SERR* pin must implement this bit, also bit[6] must be on to report address parity errors when SERR* is “1”. 1: Device must take its normal action when a parity error id detected. 0: Device must set its parity errors detection status bit (bit[15] in the status register) when an error occurred but does not assert PERR* and continues its normal operation (Default).
D7	WCC	Wait Cycle Control	Wait Cycle Control (RO) Fixed to “0” since the address/data are not stepped.
D6	PER	Parity Error Response	Parity Error Response This bit controls the device’s response to Parity Error. 1: Enable SERR* driver 0: Disable SERR* driver (Default)
D5	VGA	VGA Palette Snoop	VGA Palette Snoop (RO) Fixed to “0” since this module is not a VGA device.
D4	MW1	Memory Write and Invalidate Enable	Memory Write and Invalidate Enable
D3	SP	Special Cycle	Special Cycle (RO) Fixed to “0” since this module does not respond to special cycles.
D2	BM	Bus Master	Bus Master Set this bit to “1” to operate this module as a PCI bus master.
D1	MS	Memory Space	Memory Space (RO) Fixed to “0” since this module does not have memory space.
D0	IOS	I/O Space	I/O Space Set this bit to “1” to support access to Bus Master I/O registers.

Figure 6.2.3 Command Register

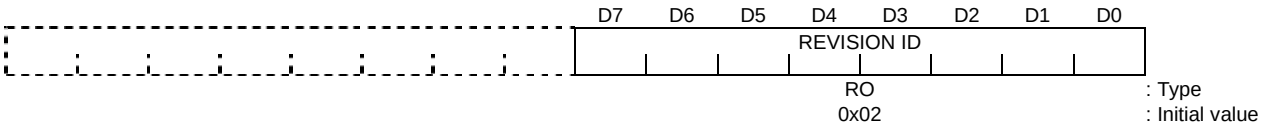
6.2.2.4 Status Register (0x06 – 0x07)

D15	D14	D13	D12	D11	D10 D9	D8	D7	D6 D5	D4	D3 D0	
DPE	SSE	RMA	RTA	STA	DECT	DPD	FBBC	Reserved	CAP	Reserved	
RWC	RWC	RWC	RWC	RWC	RO	RWC	RO	RO	RO	RO	: Type
0	0	0	0	0	01	0	1	00	1	0x00	: Initial value

Bits	Mnemonic	Field Name	Description
D15	DPE	Detected Parity Error	Detected Parity Error (RWC) This bit must be set by the device whenever it detects a parity error, even if parity error handling is disabled (as controlled by bit[6] in the Command register bit). 1: Reset 0: No action (Default)
D14	SSE	Signaled System Error	Signaled System Error (RWC) This bit must be set whenever the device asserts SERR*. Devices who will never assert SERR* do not need to implement this bit. 1: SERR* asserted 0: No SERR* asserted (Default)
D13	RMA	Received Master Abort	Received Master Abort (RWC) Is set to “1” when PCI master access performed by this module ends with a master abort.
D12	RTA	Received Target Abort	Received Target Abort (RWC) Is set to “1” when PCI master access performed by this module ends with a target abort.
D11	STA	Signaled Target Abort	Signaled Target Abort (RWC) Is set to “1” when a target abort occurs while this module is the PCI target.
D10:D9	DECT	DEVSEL Timing	DEVSEL Timing (RO) Specifies DEVSEL response timing. These bits are fixed to “01b” since this module responds using the medium DEVSEL timing.
D8	DPD	Master Data Parity Detected	Master Data Parity Detected (RWC) This bit is only implemented by Bus master. It is set when three conditions are met: 1) The bus agent asserted PERR* itself on a read or observed PERR* asserted on a write; 2) The agent setting the bit acted as the bus master for the operation in which the error occurred; and 3) The Parity Error Respond bit (Command Register’s bit[6]) is set 1: Reset 0: No action (Default)
D7	FBBC	Fast Back-to-Back Capable	Fast Back-to-Back Capable (RO) This bit indicates whether or not target is capable of accepting fast back-to-back transactions when the transactions are not to the same agent. This bit is fixed to “1” since the device can accept these transactions.
D6:D5		Reserved	
D4	CAP	Capabilities	Capabilities (RO) Set this bit to “1” to support PCI Power Management. This function is supported by default.
D3:D0		Reserved	

Figure 6.2.4 Status Register

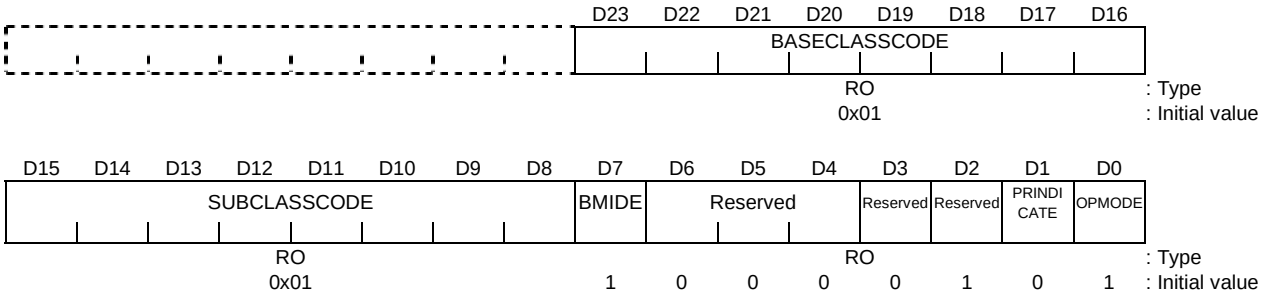
6.2.2.5 Revision ID Register (0x08)



Bits	Mnemonic	Field Name	Description
D7:D0	REVISION ID	Revision ID	Revision ID This value 0x02 represents the revision.

Figure 6.2.5 Revision ID Register

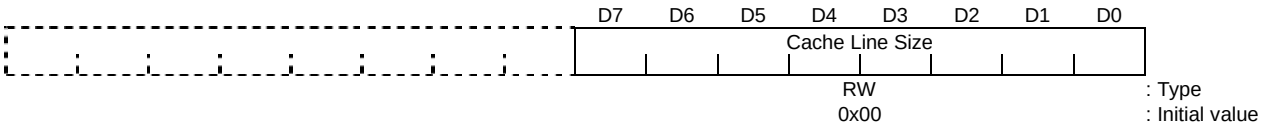
6.2.2.6 Class Code Register (0x09 – 0x0B)



Bits	Mnemonic	Field Name	Description
D23:D16	BASECLASSCODE	Base Class Code	Base Class Code This value 0x01 indicates that the base class is a storage device.
D15:D8	SUBCLASSCODE	Sub Class Code	Sub Class Code This value 0x01 indicates that the subclass is an enhanced IDE.
D7	BMIDE	BMIDE	BMIDE Fix this bit to “1” to support the Bus Master IDE capability.
D6:D4		Reserved	
D3		Reserved	Reserved Fixed to “0”.
D2		Reserved	Reserved Fix this bit to same value as D0. Fixed to “1”.
D1	PRINDICATE	PRINDICATE	PRINDICATE (Programmable Indicator) Fixed to “0” since this module not support switchover between the Native-PCI mode and the Compatibility mode.
D0	OPMODE	OPMODE	OPMODE (Operating Mode) Fixed to “1” since this module only support the Native-PCI mode. * Refer to the PCI IDE Controller Specification Rev. 1.0 for information regarding these modes.

Figure 6.2.6 Class Code Register

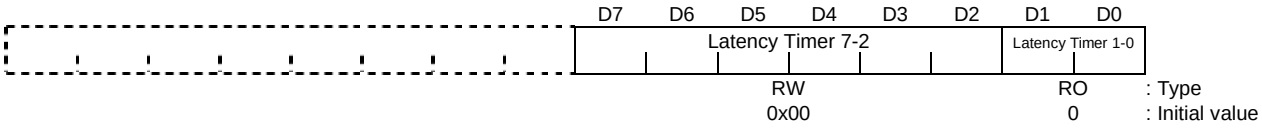
6.2.2.7 Cache Line Size Register (0x0C)



Bits	Mnemonic	Field Name	Description
D7:D0	Cache Line Size	Cache Line Size	Cache Line Size This register specifies the system cache line size in units of DWORDs.

Figure 6.2.7 Cache Line Size Register

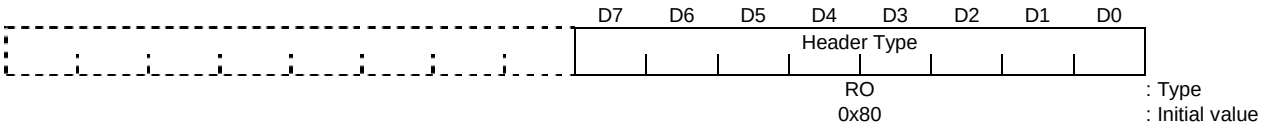
6.2.2.8 Latency Timer Register (0x0D)



Bits	Mnemonic	Field Name	Description
D7:D0	Latency Timer	Latency Timer	Latency Timer This register specifies, in units of PCI bus clocks, the value of the Latency Timer for this PCI bus master.

Figure 6.2.8 Latency Timer Register

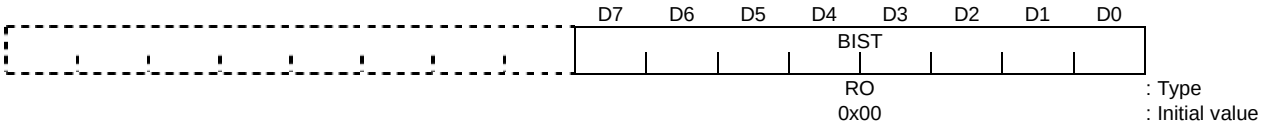
6.2.2.9 Header Type Register (0x0E)



Bits	Mnemonic	Field Name	Description
D7:D0	Header Type	Header Type	Header Type This value 0x80 represents a Multifunction device.

Figure 6.2.9 Header Type Register

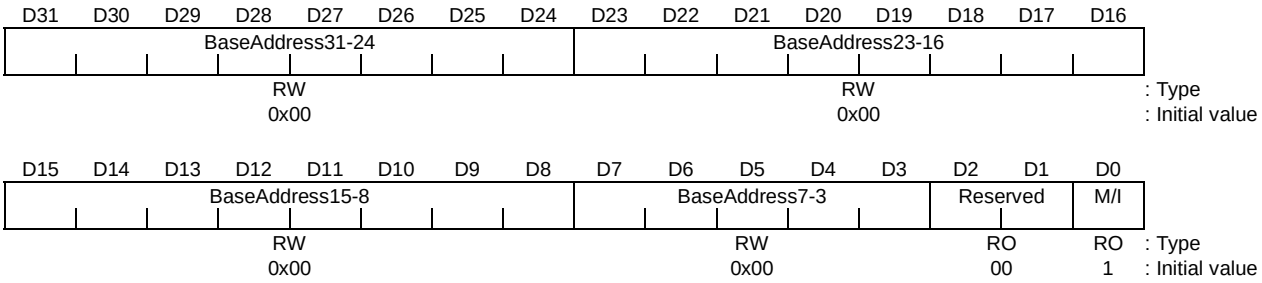
6.2.2.10 Built-in Self Test Register (0x0F)



Bits	Mnemonic	Field Name	Description
D7:D0	BIST	Built-in Self Test	Built-in Self Test Fixed to “0” since BIST is not supported.

Figure 6.2.10 Built-in Self Test Register

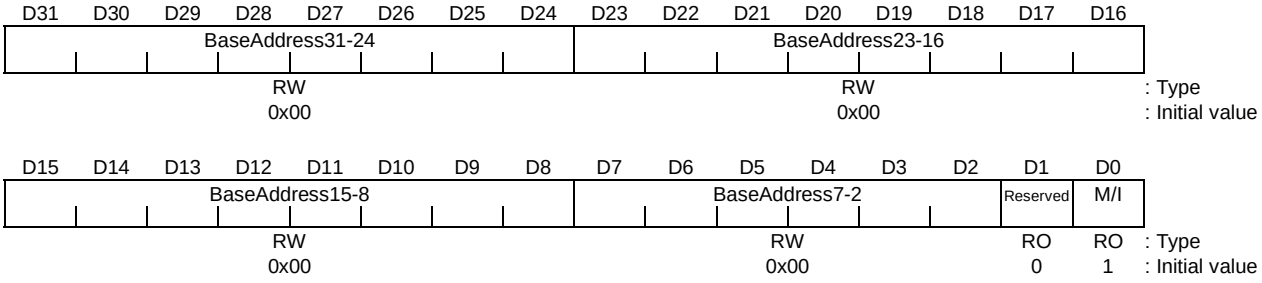
6.2.2.11 Command IO Base Address Register (0x10 – 0x13)



Bits	Mnemonic	Field Name	Description
D31:D3	Base Address	Base Address	Base Address This field is the upper 29 bits of the Command IO Register base address. Don't set D3 bit to 1. (Refer to Chapter 14 14.10)
D2:D1		Reserved	Reserved Fixed to"0" since the IO address is opened in 8-byte units.
D0	M/I	IO space Indicator	IO space Indicator Fixed to "1" since the base address register is used for I/O.

Figure 6.2.11 Command IO Base Address Register

6.2.2.12 Control IO Base Address Register (0x14 – 0x17)



Bits	Mnemonic	Field Name	Description
D31:D2	Base Address	Base Address	Base Address This field is the upper 30 bits of the Control IO Register base address.
D1		Reserved	Reserved Fixed to"0" since the IO address is opened in 4-byte units.
D0	M/I	IO space Indicator	IO space Indicator Fixed to "1" since the base address register is used for I/O.

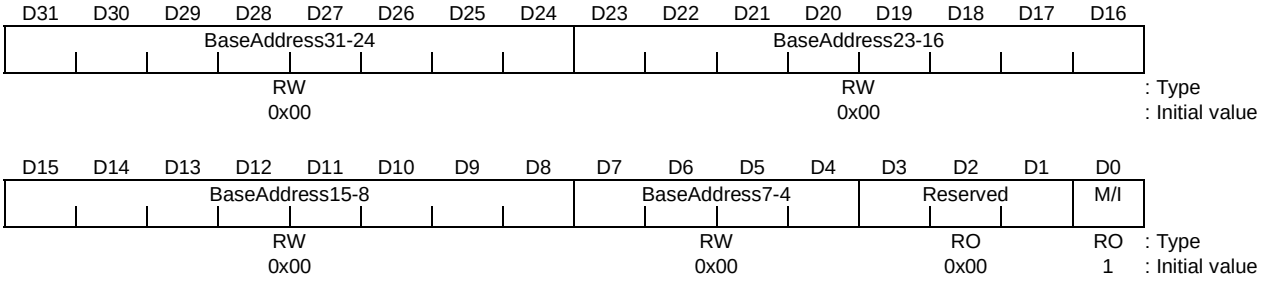
Figure 6.2.12 Control IO Base Address Register

6.2.2.13 Bus Master IDE IO Base Address Register (0x20 – 0x23)

Bits	Mnemonic	Field Name	Description
D31:D4	Base Address	Base Address	Base Address This field is the upper 28 bits of the Bus Master IO Register base address.
D3:D1		Reserved	Reserved Fixed to “0” since the IO address is opened in 16-byte units.
D0	M/I	IO space Indicator	IO space Indicator Fixed to “1” since the base address register is used for I/O.

Figure 6.2.13 Bus Master IDE IO Base Address Register

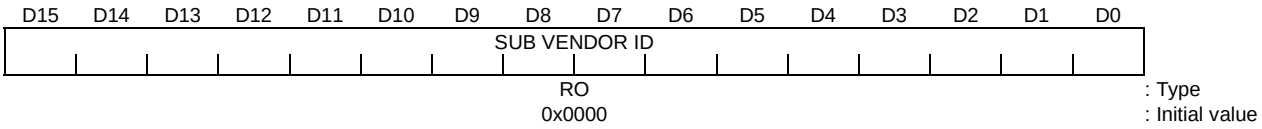
6.2.2.14 System Control IO Base Address Register (0x24 – 0x27)



Bits	Mnemonic	Field Name	Description
D31:D4	Base Address	Base Address	Base Address This field is the upper 28 bits of the System Control IO Register base address.
D3:D1		Reserved	Reserved Fixed to “0” since the IO address is opened in 16-byte units.
D0	M/I	IO space Indicator	IO space Indicator Fixed to “1” since the base address register is used for I/O.

Figure 6.2.14 System Control IO Base Address Register

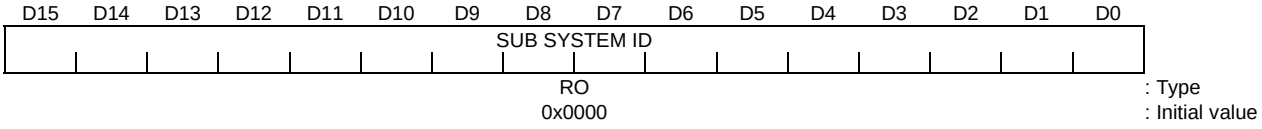
6.2.2.15 Sub Vendor ID Register (0x2C – 0x2D)



Bits	Mnemonic	Field Name	Description
D15:D0	SUB VENDOR ID	Sub System ID	Sub System ID If use Power On Loading function, please refer to 10.Loadable PCI Configuration Space.

Figure 6.2.15 Sub Vendor ID Register

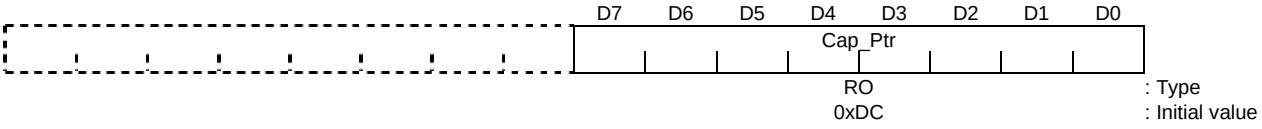
6.2.2.16 Sub System ID Register (0x2E – 0x2F)



Bits	Mnemonic	Field Name	Description
D15:D0	SUB SYSTEM ID	Sub System ID	Sub System ID If use Power On Loading function, please refer to 10.Loadable PCI Configuration Space.

Figure 6.2.16 Sub System ID Register

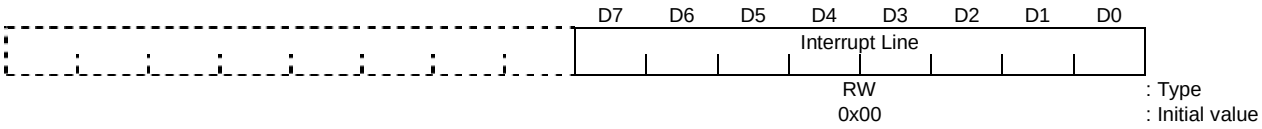
6.2.2.17 Capability Pointer Register (0x34)



Bits	Mnemonic	Field Name	Description
D7:D0	Cap_Ptr	Capability Pointer	Capability Pointer This is the address of the PCI power management register block.

Figure 6.2.17 Capability Pointer Register

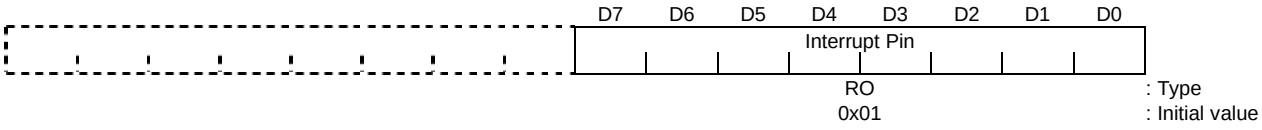
6.2.2.18 PCI Interrupt Line Register (0x3C)



Bits	Mnemonic	Field Name	Description
D7:D0	Interrupt Line	Interrupt Line	Interrupt Line These bits specify an IRQ interrupt number.

Figure 6.2.18 PCI Interrupt Line Register

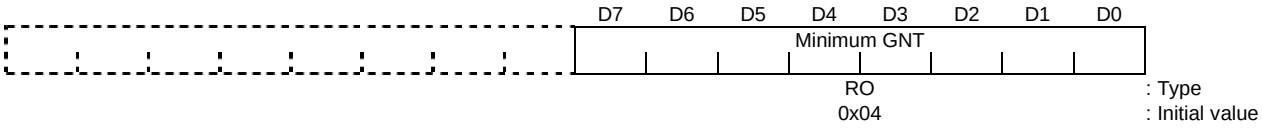
6.2.2.19 PCI Interrupt Pin Register (0x3D)



Bits	Mnemonic	Field Name	Description
D7:D0	Interrupt Pin	Interrupt Pin	Interrupt Pin This register tells which Interrupt PIN the device uses. It can be specified by the PCI Interrupt PIN Selectable Register (0x44).

Figure 6.2.19 PCI Interrupt Pin Register

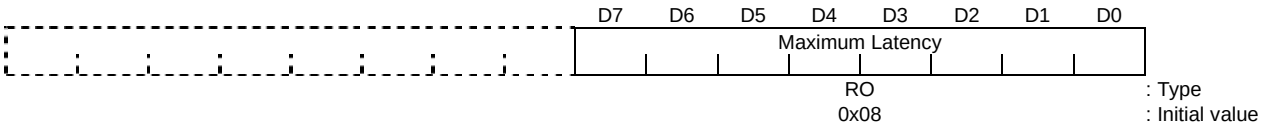
6.2.2.20 Minimum GNT Register (0x3E)



Bits	Mnemonic	Field Name	Description
D7:D0	Minimum GNT	Minimum GNT	Minimum GNT This is the minimum burst time required for this module. Specify it in units of 0.25 µs. The minimum burst time for this module is fixed to “0x04.”

Figure 6.2.20 Minimum GNT Register

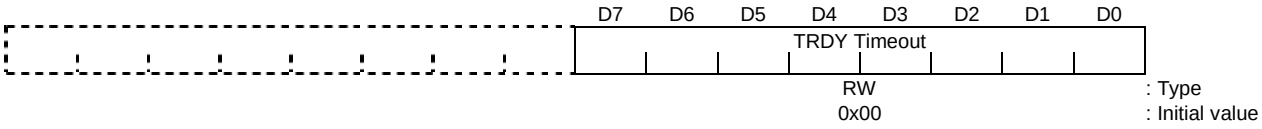
6.2.2.21 Maximum Latency Register (0x3F)



Bits	Mnemonic	Field Name	Description
D7:D0	Maximum Latency	Maximum Latency	Maximum Latency This is the maximum wait time during PCI access. Specify it in units of 0.25 µs. The maximum wait time for this module is fixed to 0x08.

Figure 6.2.21 Maximum Latency Register

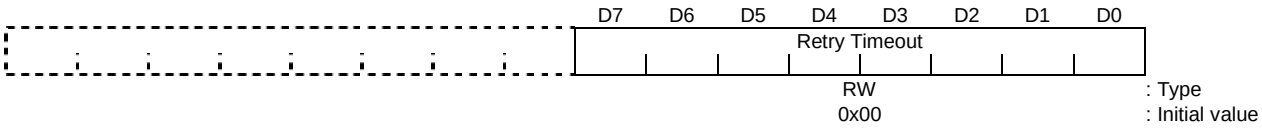
6.2.2.22 TRDY Timeout Register (0x40)



Bits	Mnemonic	Field Name	Description
D7:D0	TRDY Time out	TRDY Time out	TRDY Time out Value Sets number of PCI clocks that core as Master will wait for TRDY.

Figure 6.2.22 TRDY Timeout Register

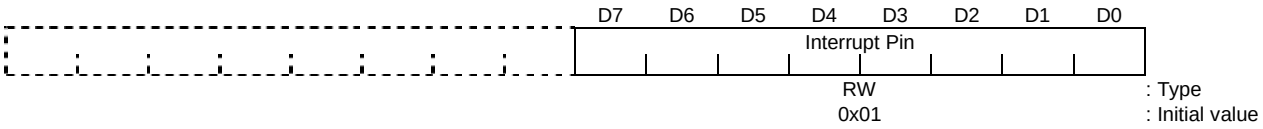
6.2.2.23 Retry Timeout Register (0x41)



Bits	Mnemonic	Field Name	Description
D7:D0	Retry Time out	Retry Time out	Retry Timeout Value Sets number of retries that the core as Master will perform.

Figure 6.2.23 Retry Timeout Register

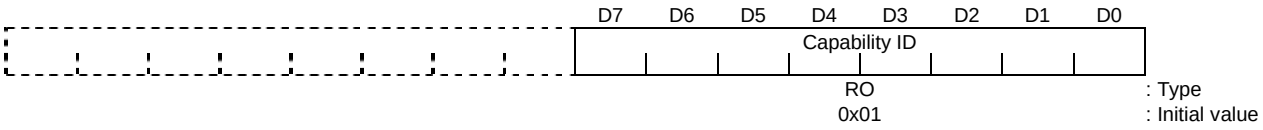
6.2.2.24 PCI Interrupt Pin Selectable Register (0x44)



Bits	Mnemonic	Field Name	Description
D7:D0	Interrupt Pin	Interrupt Pin	Interrupt Pin These bits set a PCI interrupt. 0x01: INTA* 0x02: INTB* Others: Default (INTA*)

Figure 6.2.24 PCI Interrupt Pin Selectable Register

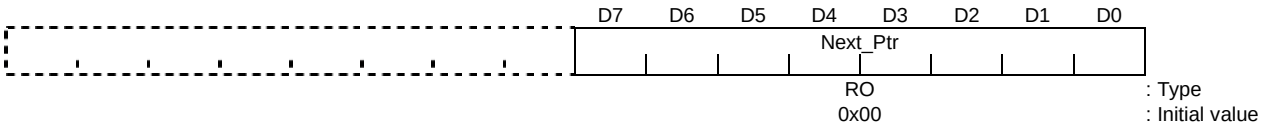
6.2.2.25 Capability ID Register (0xDC)



Bits	Mnemonic	Field Name	Description
D7:D0	Capability ID	Capability ID	Capability ID Fixed to “1.” Represents the PCI power management register block.

Figure 6.2.25 Capability ID Register

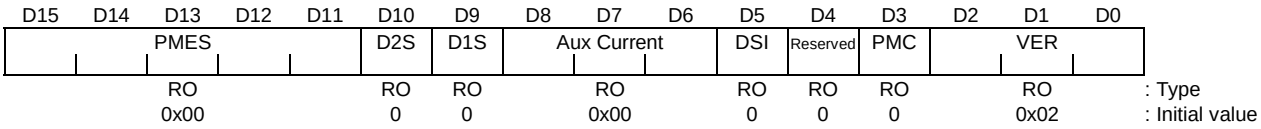
6.2.2.26 Next Item Pointer Register (0xDD)



Bits	Mnemonic	Field Name	Description
D7:D0	Next_Ptr	Next Item Pointer	Next Item Pointer Fixed to “0x00” since no next block exists.

Figure 6.2.26 Next Item Pointer Register

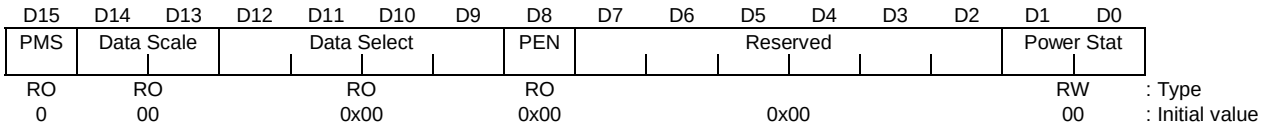
6.2.2.27 Power Management Capabilities PMC (0xDE – 0xDF)



Bits	Mnemonic	Field Name	Description
D15:D11	PMES	PME Support	PME Support Set these bits to “0” since PME* is not supported.
D10	D2S	D2 Support	D2 Support Set this bit to “0” since D2 is not supported.
D9	D1S	D1 Support	D1 Support Set this bit to “0” since D1 is not supported
D8:D6	Aux Current	Aux Current	Aux Current These bits set the auxiliary current value required for the 3.3 V auxiliary power supply. Fixed to “0” since PME* is not supported.
D5	DSI	Device Specific Initialization	Device Specific Initialization
D4		Reserved	
D3	PMC	PME Clock	PME Clock Clear this bit to “0” since PCI clock is not required to assert PME*.
D2:D0	VER	Version	Version These bits set a version number, which is “0x02” indicating that the module conforms to PCI Power Management Interface Specifications 1.1.

Figure 6.2.27 Power Management Capabilities PMC

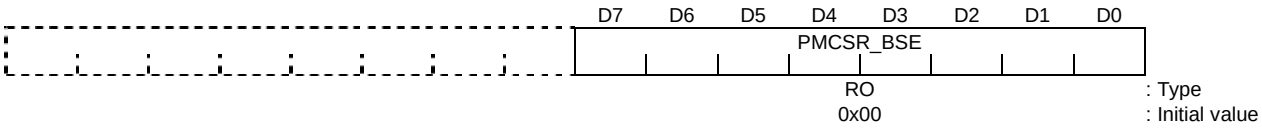
6.2.2.28 Power Management Control/Status Register PMCSR (0xE0 – 0xE1)



Bits	Mnemonic	Field Name	Description
D15	PMS	PME Status	PME Status Fixed to “0” since PME* is not supported.
D14:D13	Data Scale	Data Scale	Data Scale Fixed to “0x00 since Data scale is not supported.
D12:D9	Data Select	Data Select	Data Select Fixed to “0x00 since Data select is not supported.
D8	PEN	PME Enable	PME Enable Fixed to “0” since PME* is not supported.
D7:D2		Reserved	
D1:D0	Power Stat	Power Status	Power Status The OS uses these bits to set the current device status. 00: D0 11: D3

Figure 6.2.28 Power Management Control/Status Register PMCSR

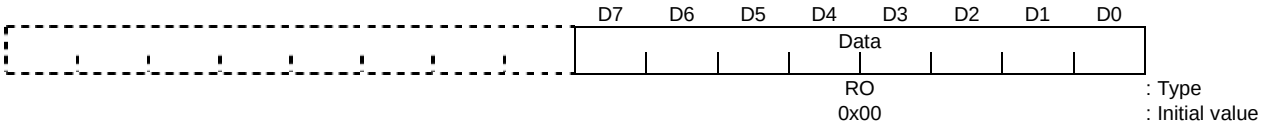
6.2.2.29 PCI to PCI Bridge Support Extension Register PMCSR BER (0xE2)



Bits	Mnemonic	Field Name	Description
D7:D0	PMCSR_BSE	MCSR_BSE	MCSR_BSE Set this bit to “0” since this module is not a PCI-PCI bridge.

Figure 6.2.29 PCI to PCI Bridge Support Extension Register PMCSR BER

6.2.2.30 Data (0xE3)



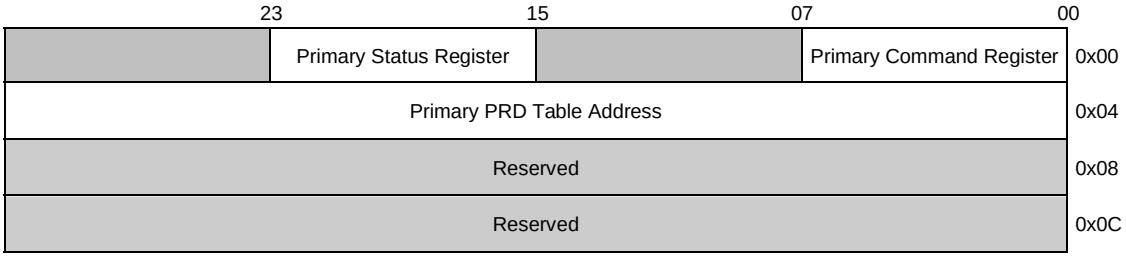
Bits	Mnemonic	Field Name	Description
D7:D0	Data	Data	Data Fixed to “0” since the Data Select is not supported.

Figure 6.2.30 Data

6.2.3 Bus Master IDE I/O Register Map

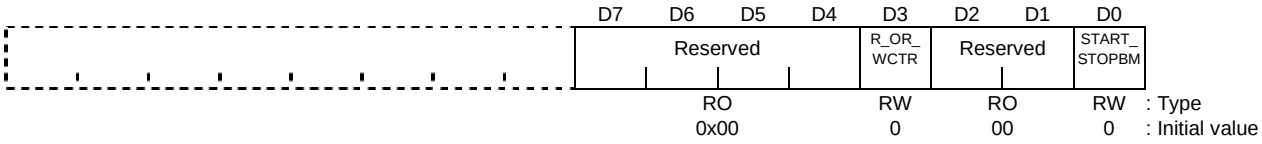
These registers are accessed by referring to the Bus Master IDE I/O base address that is set by a configuration register as an offset to the accessed registers. The I/O register base address must first be set before accessing these registers. The Bus Master IDE Register complies with Revision 1.0 of the Programming Interface for Bus Master IDE Controller.

All bus master IDE IO space registers can be accessed as 8-bit, 16-bit, 32-bit quantities.



6.2.4 Bus Master IDE I/O Register Details

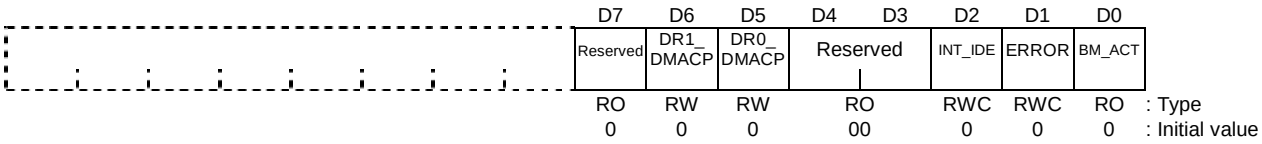
6.2.4.1 Command Register (0x00)



Bits	Mnemonic	Field Name	Description
D7:D4		Reserved	
D3	R_OR_WCTR	R_OR_WCTR	R_OR_WCTR (Read or Write Control) Specifies the direction of bus master transfers. Set this bit to “0” for PCI-to-IDE transfer (write operation). Set this bit to “1” for IDE-to-PCI transfer (read operation). Do not write to this bit during bus master operation (when bit D0 = 1).
D2:D1		Reserved	
D0	START_ STOPBM	START_STOPBM	START_STOPBM (Start/Stop Bus Master) Set this bit to “1” to start bus master transfer. Set this bit to “0” to stop the transfer. If this bit is set to “0” during bus master operation, the transfer in progress is aborted and does not resume. To perform successive transfers, temporarily set this bit to “0” then set it back to “1” again.

Figure 6.2.31 Command Register

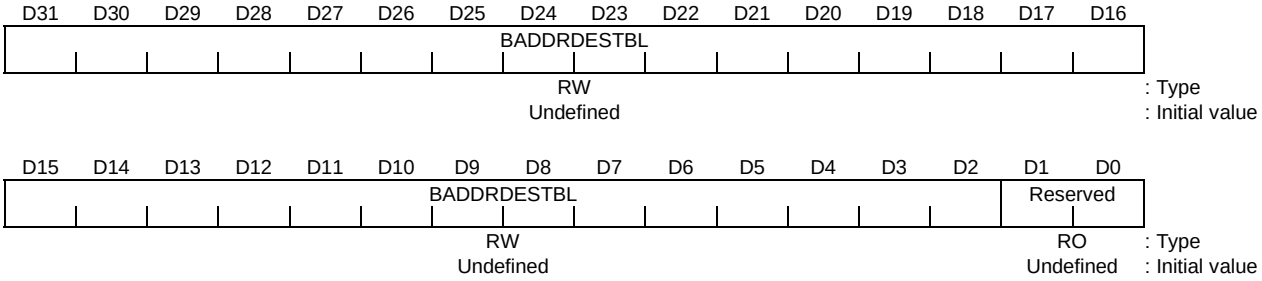
6.2.4.2 Status Register (0x02)



Bits	Mnemonic	Field Name	Description
D7		Reserved	Reserved Fixed to “0”.
D6	DR1_DMACP	DR1_DMACP	DR1_DMACP [Drive1 (Slave) DMA Capable] Is set to “1” when a slave drive can perform DMA.
D5	DR0_DMACP	DR0_DMACP	DR0_DMACP [Drive0 (Master) DMA Capable] Is set to “1” when the master drive can perform DMA.
D4:D3		Reserved	
D2	INT_IDE	INT_IDE	INT_IDE (RWC) [Interrupt] Is set to “1” when a transition of the IDE_IRQ signal to the High state is detected. When this bit is read as “1”, all data transferred from the drive is visible in system memory. This bit is cleared by writing “1” to it. (This controller’s interrupt is not cleared at this point. Software will clear the IDE device interrupt.)
D1	ERROR	ERROR	ERROR (RWC) Is set to “1” when an error occurs during data transfer to or from the system memory. This bit is cleared by writing a “1” to it. Refer to the status register in the configuration space for details about the error.
D0	BM_ACT	BM_ACT	BM_ACT(RO) [Bus Master IDE Active] Is set to “1” when the command register start bit is set to “1.” This bit is set to “0” either when the start bit is set to “0” (aborted) or when the last transfer in the current cycle is executed and the area descriptor EOT is set (terminated normally).

Figure 6.2.32 Status Register

6.2.5 Descriptor Table Pointer Register (0x04 – 0x07)



Bits	Mnemonic	Field Name	Description
D31:D2	BADDRDESTBL	BADDRDESTBL	BADDRDESTBL Specifies the upper 30 bits of the PRD base address. This address must be specified in 4-Byte alignment. The Descriptor Table must not cross a 64K boundary in memory.
D1:D0		Reserved	

Figure 6.2.33 Descriptor Table Pointer Register

6.2.6 PRD

The PRD (Physical Region Descriptor) is a table that is used to define the physical memory area used in bus master transfers. It is provided for handling situations where the memory area can only be used in small divided sections when performing bus master transfer. The minimum configuration unit of a PRD is two double-words, with each PRD located at contiguous addresses so that they can be read by the hardware. The software prepares PRDs in a memory area.

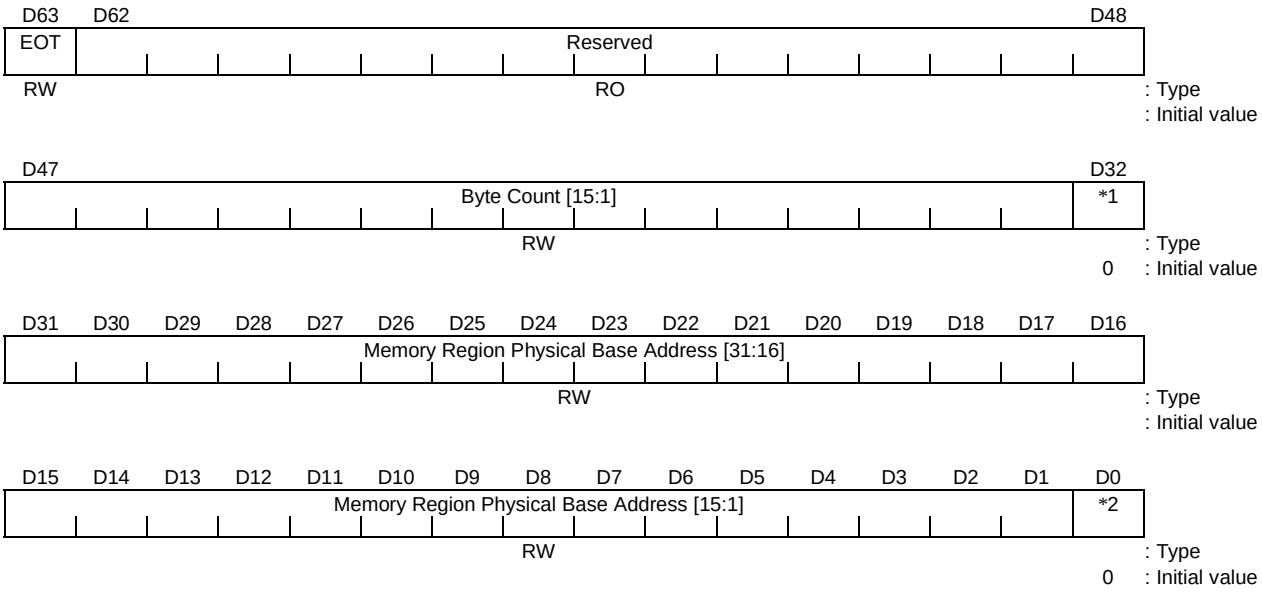
* Refer to “Programming Interface for Bus Master IDE Controller Rev. 1.0” for more information regarding PRD.

6.2.6.1 PRD Table

The PRD table has its PRD data arranged in contiguous locations beginning with the address specified by the base address. One PRD consists of two double-words (64 bits). The base address is specified by an I/O register.

Offset	
0x00	PRD
0x08	PRD
0x10	PRD
0x18	PRD (Last data)

6.2.6.2 PRD Configuration



*1: Byte Count[0]

*2: Memory Region Physical Base Address[0]

Bits	Mnemonic	Field Name	Description
D63	EOT	EOT	EOT This bit designates the end of the table. Set this bit to “1” for the last entry of the table. The DMA controller checks this bit and terminates DMA processing when it is “1.”
D62:D48		Reserved	
D47:D33	Byte Count	Byte Count	Byte Count [15:1] These bits specify the size of the memory area. The least significant bit of this byte count must be “0” since the address needs to be word-aligned.
D31:D1	Memory Region Physical Base Address	Memory Region Physical Base Address	Memory Region Physical Base Address [31:1] These bits specify the start address of the memory area in which to store data during DMA transfer. The least significant bit of this address must be “0” since the address needs to be word-aligned.

Figure 6.2.34 PRD Configuration

6.2.7 ATA I/O Register Map

ATA I/O registers are accessible at 8 bits except for the Data register.

Data registers are accessible at 16 bits.

Command Block Register and Control Block Register are accessed according to the offset from the base address specified by the Command I/O base address register and the Control I/O address register of the PCI Configuration Register. The I/O register base address must first be set before accessing these registers. These Data register is accessed as a 16-bit register for PIO transfers (except for ECC bytes). All other registers are accessed as 8-bit quantities.

6.2.7.1 Command Block Registers

GOKU-S supports native-PCI mode only.

Table 6.2.2 Command Block Register List

Offset Address	Size	Register	R/W
0x00	2	Data	RW
0x01	1	Error/Features	RW
0x02	1	Sector Count	RW
0x03	1	Sector Number	RW
0x04	1	Cylinder Low	RW
0x05	1	Cylinder High	RW
0x06	1	Device/Head	RW
0x07	1	Status/Command	RW

6.2.7.2 Control Block Registers

Table 6.2.3 Control Block Register List

Offset Address	Size	Register	R/W
0x00	1	Reserved (Fixed to 0xFF)	RO
0x01	1	Reserved (Fixed to 0xFF)	RO
0x02	1	Alternate Status/Device control	RW
0x03	1	Reserved (Fixed to 0xFF)	RO

Alternate Status/Device control register are accessed as 8-bit quantities

6.2.8 System Control I/O Register Map

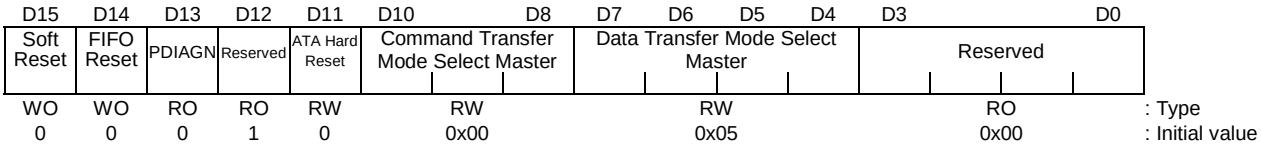
System Control I/O Register is accessed according to the offset from the base address specified by the System Control I/O base address register of the PCI Configuration Register. The I/O register base address must first be set before accessing these registers. These registers are 16-bit access only.

Table 6.2.4 System Control I/O Register List

Offset Address	Size	Register	R/W
0x00	2	System Control1	RW
0x02	2	System Control2	RW
0x04	2	Transfer Word Count1	RW
0x06	2	Transfer Word Count2	RW
0x08	2	ATAPI Packet Write	WO
0x0A	2	Sector Count	RW
0x0C	2	Sector Count Control	RW
0x0E	2	Reserved (fixed to “0”)	RO

6.2.9 System Control I/O Register Details

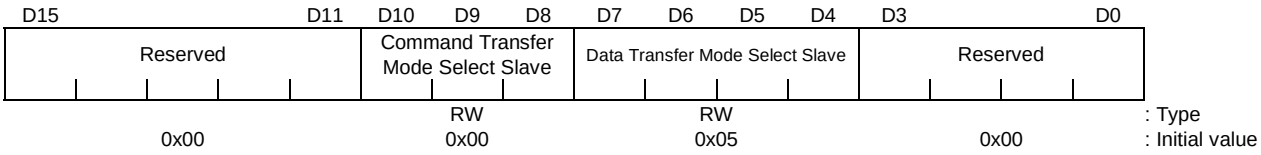
6.2.9.1 System Control 1 Register (0x00 – 0x01)



Bits	Mnemonic	Field Name	Description
D15	Soft Reset	Soft Reset	Soft Reset The CPU can apply a reset by setting this bit.
D14	FIFO Reset	FIFO Reset	FIFO Reset Executes FIFO reset. FIFO referred to here is 16-bit × 8-stage FIFO placed between the HDD and SRAM that controls/absorbs data transfer timing.
D13	PDIAGN	PDIAGN	PDIAGN(RO) Can monitor the PDIAGN signal on the Host Bus.
D12		Reserved	
D11	ATA Hard Reset	ATA Hard Reset	ATA Hard Reset Setting this bit makes it possible to create a reset signal that is sent to the ATA Bus.
D10:D8	PIO Transfer Mode Select Master	PIO Transfer Mode Select for Master Device	PIO Transfer Mode Select for Master Device Selects the PIO command transfer mode that this system uses. 0x00: PIO Mode 0 0x01: PIO Mode 1 0x02: PIO Mode 2 0x03: PIO Mode 3 0x04: PIO Mode 4
D7:D4	DMA Transfer Mode Select Master	DMA Transfer Mode Select for Master Device	DMA Transfer Mode Select for Master Device Selects the DMA transfer mode that this system uses. 0x00: Reserved 0x01: Reserved 0x02: Reserved 0x03: Reserved 0x04: Reserved 0x05: Multi Word DMA Mode 0 0x06: Multi Word DMA Mode 1 0x07: Multi Word DMA Mode 2 0x08: Ultra DMA Mode 0 0x09: Ultra DMA Mode 1 0x0A: Ultra DMA Mode 2 0x0B: Ultra DMA Mode 3 0x0C: Ultra DMA Mode 4
D3:D0		Reserved	

Figure 6.2.35 System Control 1 Register

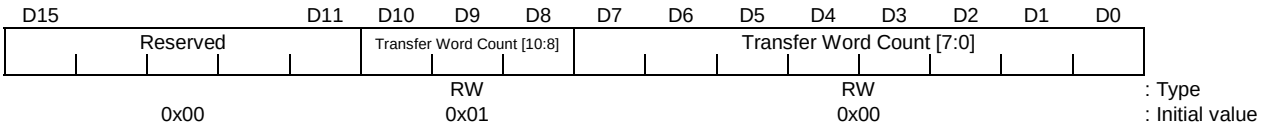
6.2.9.2 System Control 2 Register (0x02 – 0x03)



Bits	Mnemonic	Field Name	Description
D15:D11		Reserved	
D10:D8	PIO Transfer Mode Select Slave	PIO Transfer Mode Select for Slave Device	PIO Transfer Mode Select for Slave Device Selects the PIO command transfer mode that this system uses. 0x00: PIO Mode 0 0x01: PIO Mode 1 0x02: PIO Mode 2 0x03: PIO Mode 3 0x04: PIO Mode 4
D7:D4	DMA Transfer Mode Select Slave	DMA Transfer Mode Select for Slave Device	DMA Transfer Mode Select for Slave Device Selects the Data transfer mode that this system uses. 0x00: Reserved 0x01: Reserved 0x02: Reserved 0x03: Reserved 0x04: Reserved 0x05: Multi Word DMA Mode 0 0x06: Multi Word DMA Mode 1 0x07: Multi Word DMA Mode 2 0x08: Ultra DMA Mode 0 0x09: Ultra DMA Mode 1 0x0A: Ultra DMA Mode 2 0x0B: Ultra DMA Mode 3 0x0C: Ultra DMA Mode 4
D3:D0		Reserved	

Figure 6.2.36 System Control 2 Register

6.2.9.3 Transfer Word Count Register 1 (0x04 – 0x05)

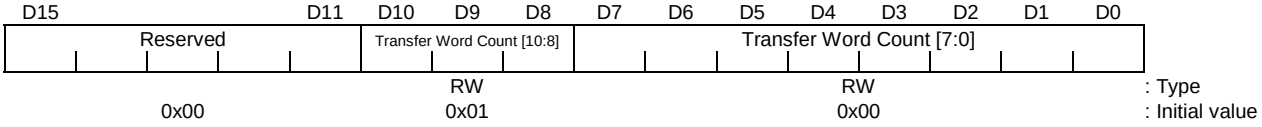


This register sets the word count that is transferred to a master device. When the size of one sector transferred to a device is 512 bytes, it is not necessary to change this register from the default sate.

Note : 1 word = 16 bits

Figure 6.2.37 Transfer Word Count Register 1

6.2.9.4 Transfer Word Count Register 2 (0x06 – 0x07)

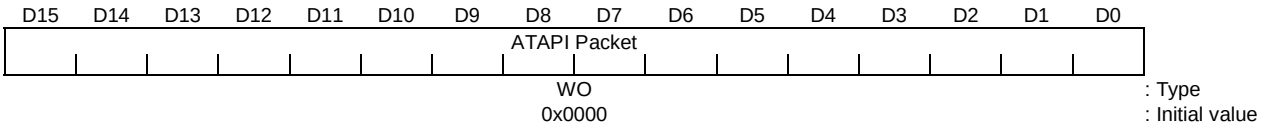


This register sets the word count that is transferred to a slave device. When the size of one sector transferred to a device is 512 bytes, it is not necessary to change this register from the default sate.

Note : 1 word = 16 bits

Figure 6.2.38 Transfer Word Count Register 2

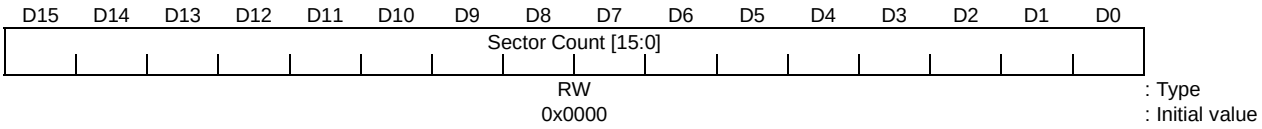
6.2.9.5 ATAPI Packet Write Register (0x08 – 0x09)



Bits	Mnemonic	Field Name	Description
D15:D0	ATAPI Packet	ATAPI Packet	ATAPI Packet Setting packet data in this register makes it possible to transmit packet data to a device. In addition, addresses (HA, CS0N, CS1N) transmitted to a device are set automatically. The following table indicates the command packet configuration. Even bytes of the following packets are written to bits 7:0 of the above ATAPI Packet Command Register. Odd bytes are written to bits 15:8. Bytes: Content 0: Instruction Code 1: Parameter 0 2: Parameter 1 3: Parameter 2 4: Parameter 3 5: Parameter 4 6: Parameter 5 7: Parameter 6 8: Parameter 7 9: Parameter 8 10: Parameter 9 11: Parameter 10

Figure 6.2.39 ATAPI Packet Write Register

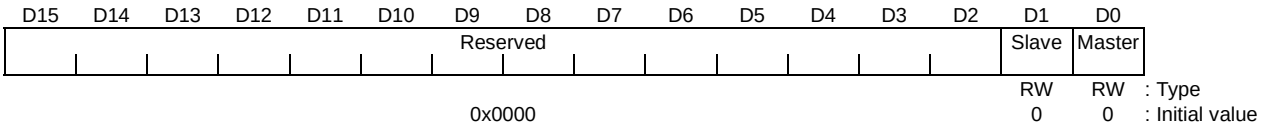
6.2.9.6 Sector Count Register (0x0A – 0x0B)



This setting can be used for determining the sector count to transfer. In addition, the Transfer Word Count determines the transfer byte count per sector. (Goku-S's sector Count is max 0xFFFF (65,535).)

Figure 6.2.40 Sector Count Register

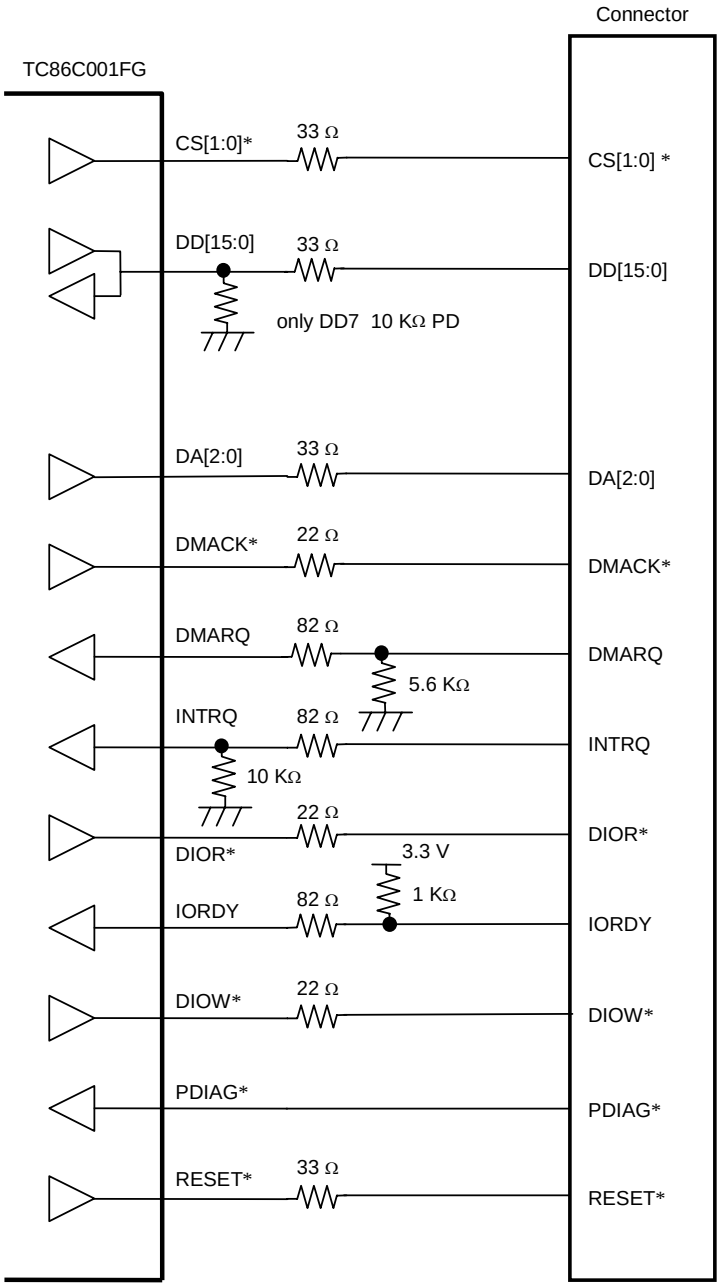
6.2.9.7 Sector Count Control Register (0x0C – 0x0D)



Bits	Mnemonic	Field Name	Description
D15:D2		Reserved	
D1	Sector Count Control Slave	Sector Count Control for Slave Device	Sector Count Control for Slave Device 0: Hardware sets GOKU-S Sector Count Register automatically (Only ATA device is useful.) 1: Software sets GOKU-S Sector Count Register
D0	Sector Count Control Master	Sector Count Control for Master Device	Sector Count Control for Master Device 0: Hardware sets GOKU-S Sector Count Register automatically (Only ATA device is useful.) 1: Software sets GOKU-S Sector Count Register

Figure 6.2.41 Sector Count Control Register

6.3 ATA Connection Example



7. USB Host Controller (Function 1)

7.1 Function Description

7.1.1 Overview

The USB Host Controller (USBHC) is compliant with the USB Specification Revision 1.1 and the OpenHCI Specification Release 1.0a. The USBHC supports both full-speed (12-Mbps) and low-speed (1.5-Mbps) data rates.

The USBHC is directed attached to the 32-bit G-Bus and operates as both a master and a slave on the G-Bus. The internal DMA controller of the USBHC is responsible for moving data between memory and the USBHC's internal FIFO.

The USB and OpenHCI specifications are available at the following URLs:

- Universal Serial Bus Specification Revsion 1.1: http://www.usb.org
- Open Host Controller Interface Specification for USB Release 1.0a: http://www.usb.org

7.1.2 Features

The USBHC has the following features:

- Compliant with the OpenHCI Specification Release 1.0a.
- Compliant with the USB Specification Revision 1.1.

  Supports both full-speed (12-Mbps) and low-speed (1.5-Mbps) USB devices.
- Operates as both a master and a slave on the G-Bus.
  - The internal DMA controller behaves as a bus master for the USB Master Block.
  - The USBHC is directly attached to the 32-bit G-Bus.
  - The USBHC provides memory byte alignment support for G-Bus memory access.
  - The bridge logic contains a four-word FIFO to allow quadword burst transfers.

7.1.3 Limitations

- The USBHC provides only quadword burst transfers.
- The USBHC performs a quadword burst transaction to transfer four words of data aligned on a quadword boundary. In other cases, the USBHC generally transfers data in single-beat transactions of one word. However, when two or more words are requested within quadword boundaries, a quadword burst read occurs.
- For memory-to-memory transfers, the USBHC must ensure that no buffer underrun or overrun error occurs during isochronous transactions. Memory speed must keep up with the isochronous data rate.
- The slave access supports only single-word (32-bit) transfers.
- The USBHC does not support legacy devices.

7.1.4 Terms

Assertion and ***assert***	refer to a signal that is active or true, independent of the voltage level (1 or 0) they represent.
Deassertion and ***deassert***	refer to a signal that is inactive or false, independent of the voltage level (1 or 0) they represent.
Word	refers to consecutive 32 bits.
Byte	refers to consecutive 8 bits.

Note that the USB specification uses these terms with the following definitions:

DWORD or ***double word***	refers to consecutive 32 bits.
Word	refers to consecutive 16 bits.
Byte	refers to consecutive 8 bits.

7.1.5 OpenHCI Specification

Refer to the OpenHCI Specification Release 1.0a for details on the host controller interface, descriptor structures and the Host Controller Communication Area (HCCA).

The USBHC supports the reporting of overcurrent conditions for USB power control. By default, the USBHC allows per-port power switching. For full descriptions of overcurrent protection and power switching mode, also see the OpenHCI Specification Release 1.0a.

7.1.6 Interrupts

The USBHC has these interrupts:

- Scheduling Overrun
- HcDoneHead Writeback
- Start of Frame
- Resume Detect
- Unrecoverable Error
- Frame Number Overflow
- Root Hub Status Change
- Ownership Change

When an event that causes a hardware interrupt occurs, the USBHC sets the corresponding bit in the *HcInterruptStatus* register. When a bit becomes set, a hardware interrupt (usbgIntB) is generated if the **MasterInterruptEnable** bit is set and the corresponding bit in the **HcInterruptEnable** bit is set.

The USB Host Controller Driver (HCD) may clear specific bits in the *HcInterruptStatus* register by writing 1 to bit positions to be cleared. The HCD may not set any of these bits.

The USBHC will never clear any of these bits.

Although the OpenHCI Specification Release 1.0a specifies the System Management Interrupt (SMI), the USBHC will never generate an SMI because the USBHC does not support legacy devices.

7.1.7 Reset

The USBHC has both hardware and software resets.

7.1.7.1 Hardware Reset

A hardware reset occurs when the FULLRST* input is asserted (Low).

- All registers are initialized to their default states.
- The Host Controller forces reset signaling on the USB bus (DP=DN=0).
- The USBHC transitions to the UsbReset state.
- Endpoint list processing and SOF token generation are disabled.
- The **FrameNumber** field of the *HcFmNumber* register is frozen.

7.1.7.2 Software Reset

A software reset is initiated when the **HostControllerReset** bit in the *HcCommandStatus* register is set.

- All registers are initialized to their default states.
- The Host Controller forces reset signaling on the USB bus (DP=DN=0).
- The USBHC transitions to the UsbSuspend state.
- The **FunctionalState** field in the *HcControl* register is set to 0x03.
- The USB HCD does not modify the **InterruptRouting** bit in the *HcControl* register.
- The USBHC does not modify the **RemoteWakeupConnected** bit in the *HcControl* register.

7.1.8 Port Power Control

The USBHC has a control signal to turn on and off external power ICs. Port power is controlled through the USBHPONx outputs (where x is 0 or 1). By default, the USBHC allows per-port power switching. Writing a 1 to the **LPSC** bit in the *HcRhStatus* register (at offset 0x50) turns on power to ports. This drives the USBHPONx outputs High.

Overcurrent conditions are detected at the USBHOCx* inputs (where x is 0 or 1). A Low on these inputs causes the **OCI** bit in the *HcRhStatus* register (at offset 0x50) to be set.

7.2 Register Description

The USB Host Controller has one PCI configuration space and OpenHCI-compliant memory addresses. The memory registers are accessed after setting the base address in the configuration register.

Table 7.2.1 Register Access Type

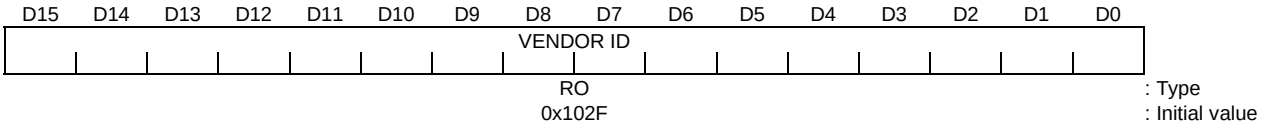
Access Type	Meaning
RO	Read-only
WO	Write-only
RW	Can both be read and written to.
RWC	Can both be read and written to. Cleared when written to.

7.2.1 Configuration Map

23	15	07	00	
Device ID		Vendor ID		0x00
Status		Command		0x04
Class Code			Revision ID	0x08
BIST	Header Type	Latency Timer	Cache Line Size	0x0C
OpenHCI memory base address				0x10
				0x14
				0x18
				0x1C
				0x20
				0x24
				0x28
Sub System ID		Sub Vendor ID		0x2C
				0x30
			Capability Pointer	0x34
				0x38
Maximum Latency	Minimum GNT	PCI Interrupt Pin	PCI Interrupt Line	0x3C
		Retry Timeout	TRDY Timeout	0x40
			PCI Interrupt Pin select	0x44
				0x48-0xD8
Power Management Capabilities		Next Item Pointer	Capability ID	0xDC
Data	PMCSR BSE	Power Management Control/Status		0xE0
				0xE4-0xFC

7.2.2 Configuration Register Details

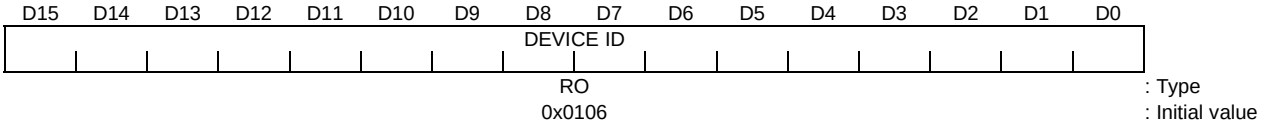
7.2.2.1 Vendor ID Register (0x00 – 0x01)



Bits	Mnemonic	Field Name	Description
D15:D0	VENDOR ID	Vendor ID	Vendor ID This value 0x102F represents the Toshiba Semiconductor Company.

Figure 7.2.1 Vendor ID Register

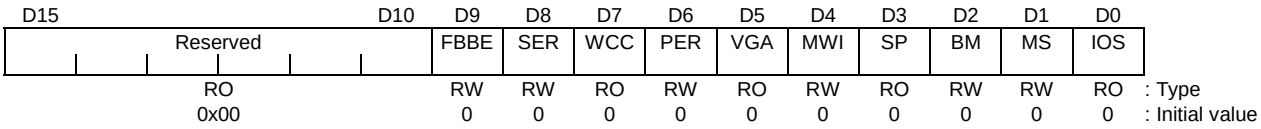
7.2.2.2 Device ID Register (0x02 – 0x03)



Bits	Mnemonic	Field Name	Description
D15:D0	DEVICE ID	Device ID	Device ID This value 0x0106 represents the USB Host Controller.

Figure 7.2.2 Device ID Register

7.2.2.3 Command Register (0x04 – 0x05)



Bits	Mnemonic	Field Name	Description
D15:D10		Reserved	
D9	FBBE	Fast Back-to-Back Enable	This bit controls whether or not a master can do fast back-to-back transactions to different devices. A value of 1 means the master is allowed to generate fast back-to-back transactions to different agents. A value of 0 means fast back-to-back transactions are only allowed to the same agent.
D8	SER	SERR Enable	SERR Enable This is the enable bit for the SERR* driver. All devices that have an SERR* pin must implement this bit, also bit[6] must be on to report address parity errors when SERR* is “1”. 1: Device must take its normal action when a parity error id detected. 0: Device must set its parity errors detection status bit (bit[15] in the status register) when an error occurred but does not assert PERR* and continues its normal operation (Default).
D7	WCC	Wait Cycle Control	Wait Cycle Control Fixed to “0” since the address/data are not stepped.
D6	PER	Parity Error Response	Parity Error Response This bit controls the device’s response to Parity Error. 1: Enable SERR* driver 0: Disable SERR* driver (Default)
D5	VGA	VGA Palette Snoop	VGA Palette Snoop Fixed to “0” since this module is not a VGA device.
D4	MWI	Memory Write and Invalidate Enable	Memory Write and Invalidate Enable
D3	SP	Special Cycle	Special Cycle Fixed to “0” since this module does not respond to special cycles.
D2	BM	Bus Master	Bus Master Set this bit to “1” to operate this module as a PCI bus master.
D1	MS	Memory Space	Memory Space Set this bit to “1” to support OpenHCI memory registers.
D0	IOS	I/O Space	I/O Space Fixed to “0” since this module does not have I/O registers.

Figure 7.2.3 Command Register

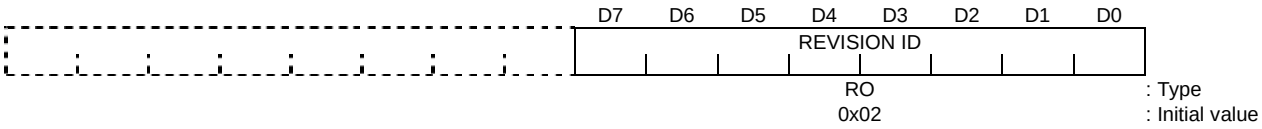
7.2.2.4 Status Register (0x06 – 0x07)

D15	D14	D13	D12	D11	D10 D9	D8	D7	D6 D5	D4	D3 D2 D1 D0	
DPE	SSE	RMA	RTA	STA	DECT	DPD	FBBC	Reserved	CAP	Reserved	
RWC	RWC	RWC	RWC	RWC	RO	RWC	RO	RO	RO	RO	: Type
0	0	0	0	0	01	0	1	00	1	0x00	: Initial value

Bits	Mnemonic	Field Name	Description
D15	DPE	Detected Parity Error	Detected Parity Error This bit must be set by the device whenever it detects a parity error, even if parity error handling is disabled (as controlled by bit[6] in the Command register bit). 1: Reset 0: No action (Default)
D14	SSE	Signaled System Error	Signaled System Error This bit must be set whenever the device asserts SERR*. Devices who will never assert SERR* do not need to implement this bit. 1: SERR* asserted 0: No SERR* asserted (Default)
D13	RMA	Received Master Abort	Received Master Abort Is set to “1” when PCI master access performed by this module ends with a master abort.
D12	RTA	Received Target Abort	Received Target Abort Is set to “1” when a target abort occurs while this module is the PCI target.
D11	STA	Signaled Target Abort	Signaled Target Abort Is set to “1” when a target abort occurs while this module is the PCI target.
D10:D9	DECT	DEVSEL Timing	DEVSEL Timing Specifies DEVSEL response timing. These bits are fixed to “01b” since this module responds using the medium DEVSEL timing.
D8	DPD	Master Data Parity Detected	Master Data Parity Detected This bit is only implemented by Bus master. It is set when three conditions are met: 1) The bus agent asserted PERR* itself on a read or observed PERR* asserted on a write; 2) T he agent setting the bit acted as the bus master for the operation in which the error occurred; and 3) T he Parity Error Respond bit (Command Register’s bit[6]) is set 1b: Reset 0b: No action (Default)
D7	FBBC	Fast Back-to-Back Capable	Fast Back-to-Back Capable This bit indicates whether or not a target is capable of accepting fast back-to-back transactions when the transactions are not to the same agent. This bit is fixed to “1” since the device can accept these transactions.
D6:D5		Reserved	
D4	CAP	Capabilities	Capabilities Set this bit to “1” to support PCI Power Management. This function is supported by default.
D3:D0		Reserved	

Figure 7.2.4 Status Register

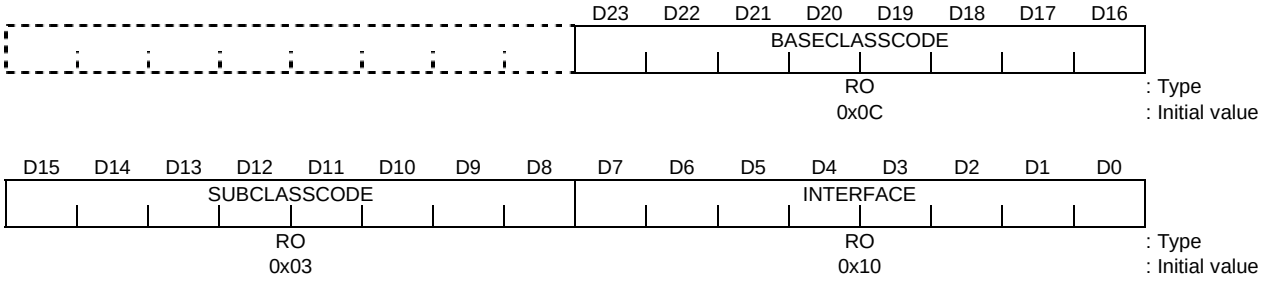
7.2.2.5 Revision ID Register (0x08)



Bits	Mnemonic	Field Name	Description
D7:D0	REVISION ID	Revision ID	Revision ID This value 0x02 represents the revision.

Figure 7.2.5 Revision ID Register

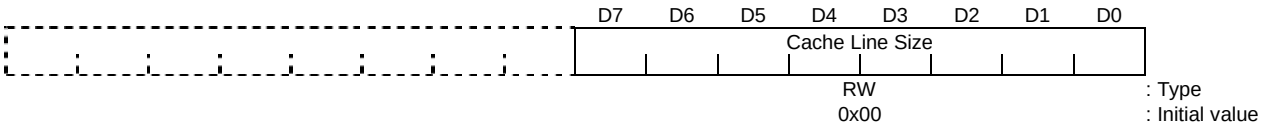
7.2.2.6 Class Code Register (0x09 – 0x0B)



Bits	Mnemonic	Field Name	Description
D23:D16	BASECLASS CODE	Base Class Code	Class Code This value 0x0C0310 represents the USB Host Controller with an OpenHCI-compliant interface.
D15:D8	SUBCLASSC ODE	Sub Class Code	
D7:D0	INTERFACE	Interface	

Figure 7.2.6 Class Code Register

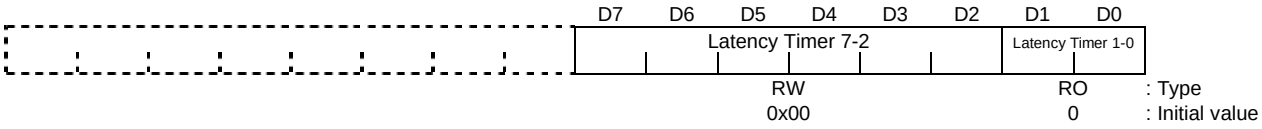
7.2.2.7 Cache Line Size Register (0x0C)



Bits	Mnemonic	Field Name	Description
D7:D0	Cache Line Size	Cache Line Size	Cache Line Size This register specifies the system cache line size in units of DWORDs.

Figure 7.2.7 Cache Line Size Register

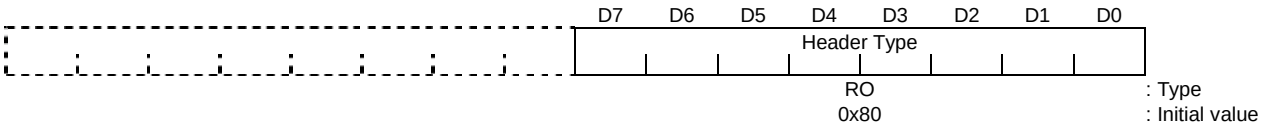
7.2.2.8 Latency Timer Register (0x0D)



Bits	Mnemonic	Field Name	Description
D7:D0	Latency Timer	Latency Timer	Latency Timer This register specifies, in units of PCI bus clocks, the value of the Latency Timer for this PCI bus master.

Figure 7.2.8 Latency Timer Register

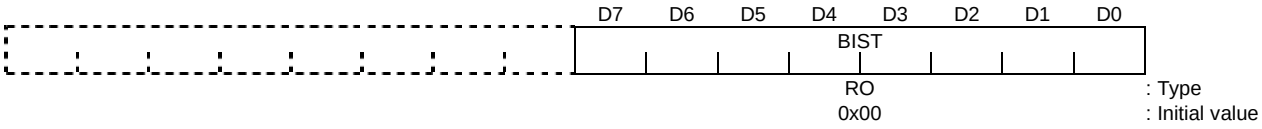
7.2.2.9 Header Type Register (0x0E)



Bits	Mnemonic	Field Name	Description
D7:D0	Header Type	Header Type	Header Type This value 0x80 represents a Multifunction device.

Figure 7.2.9 Header Type Register

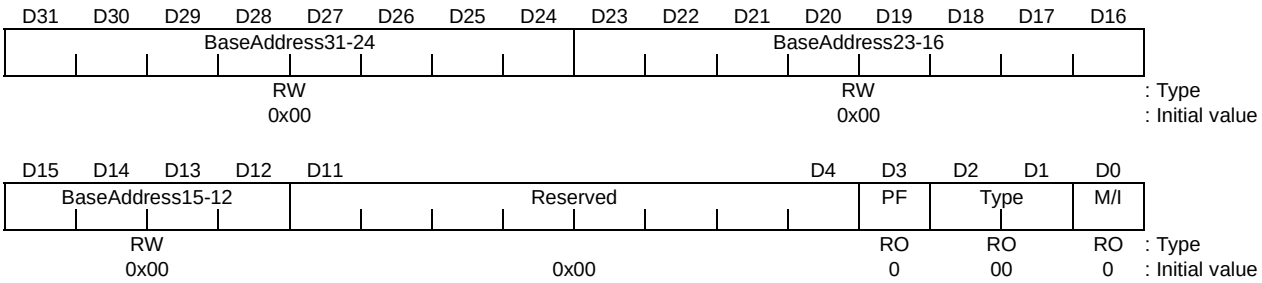
7.2.2.10 Built-in Self Test Register (0x0F)



Bits	Mnemonic	Field Name	Description
D7:D0	BIST	Built-in Self Test	Built-in Self Test Fixed to “0” since BIST is not supported.

Figure 7.2.10 Built-in Self Test Register

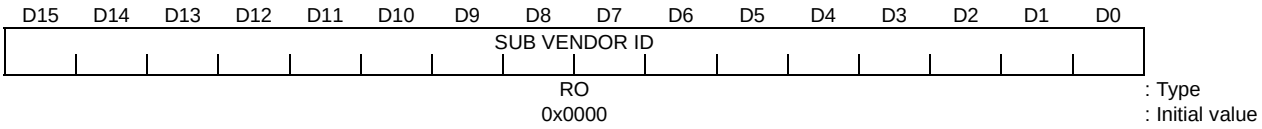
7.2.2.11 OpenHCI Memory Base Address (0x10 – 0x13)



Bits	Mnemonic	Field Name	Description
D31:D12	BaseAddress 31-24	Base Address	Base Address These bits comprise the upper 20 bits of the base address used for accessing OpenHCI registers.
D11:D4		Reserved	Reserved Fixed to “0” since OpenHCI registers need to be mapped on 4 KB boundaries.
D3	PF	Prefetchable	Prefetchable Fixed to “0” since OpenHCI registers must be allocated to a non-cacheable area.
D2:D1	Type	Type	Type Fixed to “00” since OpenHCI registers are 32-bit address spaces.
D0	M/I	Memory space Indicator	Memory space Indicator Fixed to “0” since the base address register is used for memory.

Figure 7.2.11 OpenHCI Memory Base Address

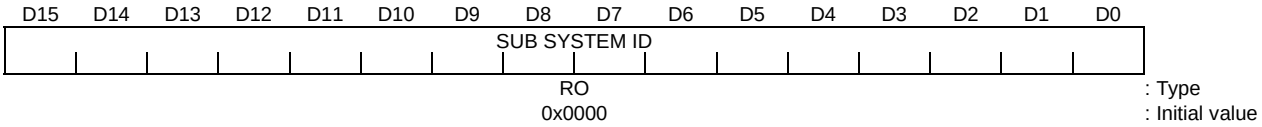
7.2.2.12 Sub Vendor ID Register (0x2C – 0x2D)



Bits	Mnemonic	Field Name	Description
D15:D0	SUB VENDOR ID	Sub Vendor ID	Sub Vendor ID If use Power On Loading function, please refer to 10.Loadable PCI Configuration Space.

Figure 7.2.12 Sub Vendor ID Register

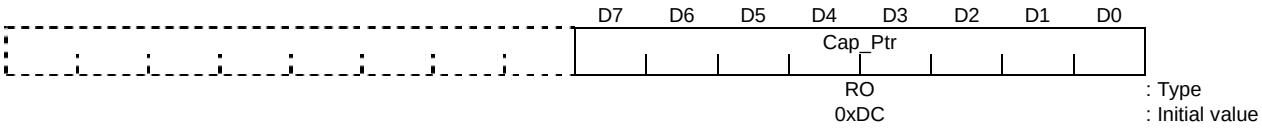
7.2.2.13 Sub System ID Register (0x2E – 0x2F)



Bits	Mnemonic	Field Name	Description
D15:D0	SUB SYSTEM ID	Sub System ID	Sub System ID If use Power On Loading function, please refer to 10.Loadable PCI Configuration Space.

Figure 7.2.13 Sub System ID Register

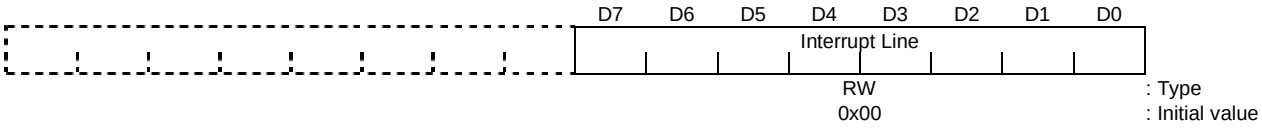
7.2.2.14 Capability Pointer Register (0x34)



Bits	Mnemonic	Field Name	Description
D7:D0	Cap_Ptr	Capability Pointer	Capability Pointer

Figure 7.2.14 Capability Pointer Register

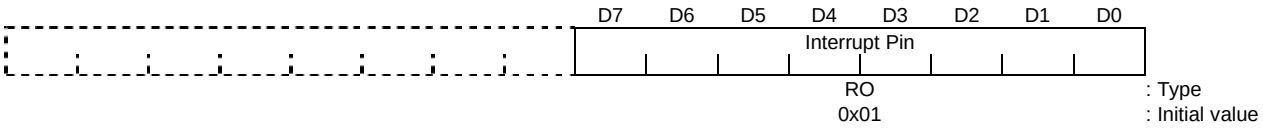
7.2.2.15 PCI Interrupt Line Register (0x3C)



Bits	Mnemonic	Field Name	Description
D7:D0	Interrupt Line	Interrupt Line	Interrupt Line These bits specify an IRQ interrupt number.

Figure 7.2.15 PCI Interrupt Line Register

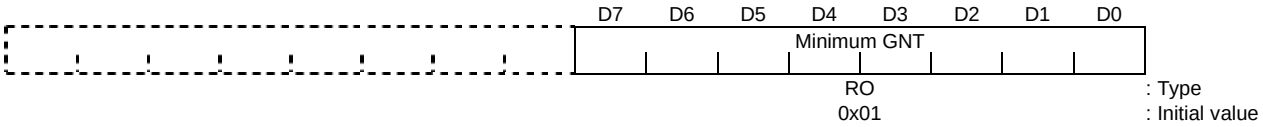
7.2.2.16 PCI Interrupt Pin Register (0x3D)



Bits	Mnemonic	Field Name	Description
D7:D0	Interrupt Pin	Interrupt Pin	Interrupt Pin This register tells which Interrupt PIN the device uses. It can be specified by the PCI Interrupt PIN Selectable Register (0x44).

Figure 7.2.16 PCI Interrupt Pin Register

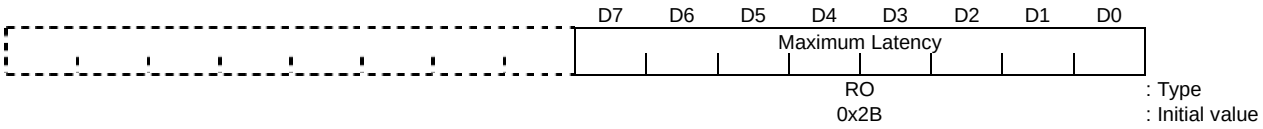
7.2.2.17 Minimum GNT Register (0x3E)



Bits	Mnemonic	Field Name	Description
D7:D0	Minimum GNT	Minimum GNT	Minimum GNT This is the minimum burst time that this module requires. Specify it in units of 0.25 µs. The minimum burst time for this module is fixed to “0x01.”

Figure 7.2.17 Minimum GNT Register

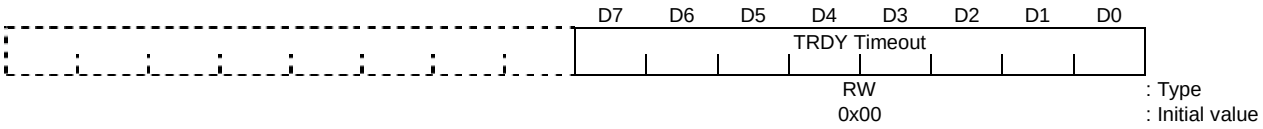
7.2.2.18 Maximum Latency Register (0x3F)



Bits	Mnemonic	Field Name	Description
D7:D0	Maximum Latency	Maximum Latency	Maximum Latency This is the maximum wait time during PCI access. Specify it in units of 0.25 µs. The maximum wait time for this module is fixed to 0x2B.

Figure 7.2.18 Maximum Latency Register

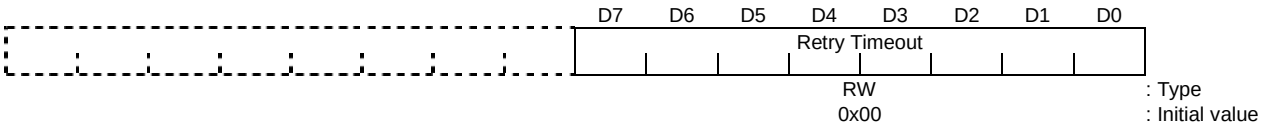
7.2.2.19 TRDY Timeout Register (0x40)



Bits	Mnemonic	Field Name	Description
D7:D0	TRDY Time out	TRDY Time out	TRDY Time out Value Sets number of PCI clocks that core as Master will wait for TRDY. Maximum value is 0xFE. If 0xFF is set, time out is invalided.

Figure 7.2.19 TRDY Timeout Register

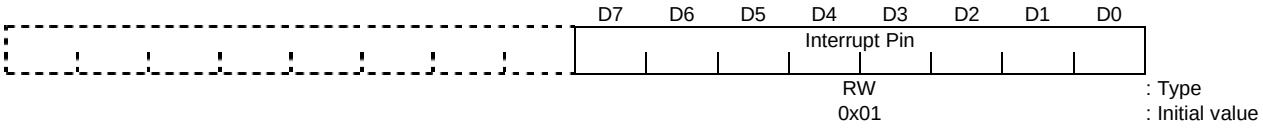
7.2.2.20 Retry Timeout Register (0x41)



Bits	Mnemonic	Field Name	Description
D7:D0	Retry Time out	Retry Time out	Retry Timeout Value Sets number of retries that the core as Master will perform.

Figure 7.2.20 Retry Timeout Register

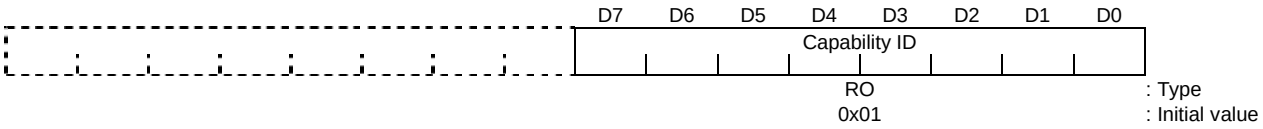
7.2.2.21 PCI Interrupt Pin Selectable Register (0x44)



Bits	Mnemonic	Field Name	Description
D7:D0	Interrupt Pin	Interrupt Pin	Interrupt Pin These bits set a PCI interrupt. 0x01: INTA* 0x02: INTB* Others: Default (INTA*)

Figure 7.2.21 PCI Interrupt Pin Selectable Register

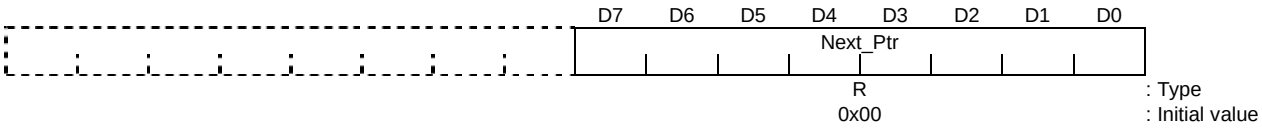
7.2.2.22 Capability ID Register (0xDC)



Bits	Mnemonic	Field Name	Description
D7:D0	Capability ID	Capability ID	Capability ID Fixed to “1.” Represents the PCI power management register block.

Figure 7.2.22 Capability ID Register

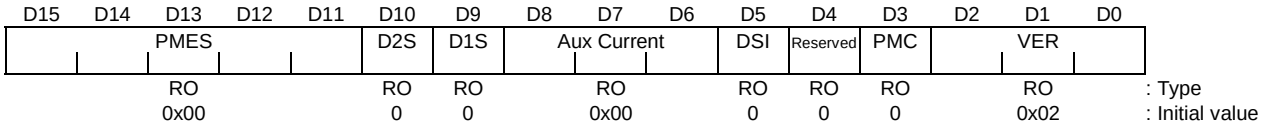
7.2.2.23 Next Item Pointer Register (0xDD)



Bits	Mnemonic	Field Name	Description
D7:D0	Next_Ptr	Next Item Pointer	Next Item Pointer Fixed to “0x00” since no next block exists.

Figure 7.2.23 Next Item Pointer Register

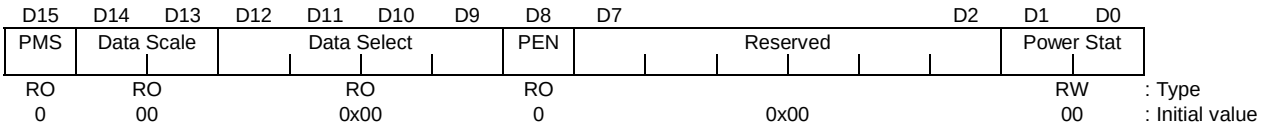
7.2.2.24 Power Management Capabilities PMC (0xDE – 0xDF)



Bits	Mnemonic	Field Name	Description
D15:D11	PMES	PME Support	PME Support Set these bits to “0” since PME* is not supported.
D10	D2S	D2 Support	D2 Support Set this bit to “0” since D2 is not supported.
D9	D1S	D1 Support	D1 Support Set this bit to “0” since D1 is not supported.
D8:D6	Aux Current	Aux Current	Aux Current These bits set the auxiliary current value required for the 3.3 V auxiliary power supply. Fixed to “0” since PME* is not supported.
D5	DSI	Device Specific Initialization	Device Specific Initialization
D4		Reserved	
D3	PMC	PME Clock	PME Clock Clear this bit to “0” since PCI clock is not required to assert PME*.
D2:D0	VER	Version	Version These bits set a version number, which is “0x02” indicating that the module conforms to PCI Power Management Interface Specifications 1.1.

Figure 7.2.24 Power Management Capabilities PMC

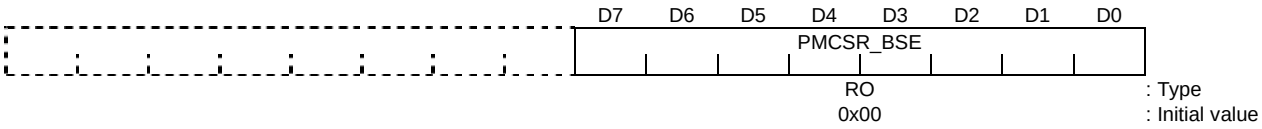
7.2.2.25 Power Management Control/Status Register PMCSR (0xE0 – 0xE1)



Bits	Mnemonic	Field Name	Description
D15	PMS	PME Status	PME Status Fixed to “0” since PME* is not supported.
D14:D13	Data Scale	Data Scale	Data Scale Fixed to “0x00” since Data scale is not supported.
D12:D9	Data Select	Data Select	Data Select These bits select a data register. Fixed to “0x00” since Data select is not supported.
D8	PEN	PME Enable	PME Enable Fixed to “0” since PME* is not supported.
D7:D2		Reserved	
D1:D0	Power Stat	Power Status	Power Status The OS uses these bits to set the current device status. 00: D0 11: D3

Figure 7.2.25 Power Management Control/Status Register PMCSR

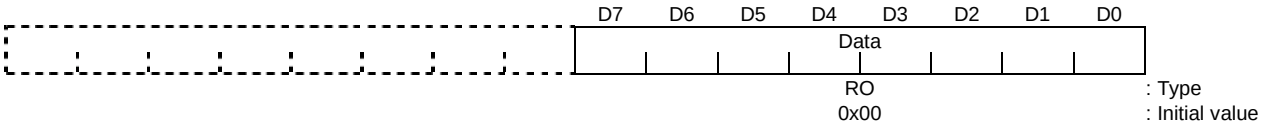
7.2.2.26 PCI to PCI Bridge Support Extension Register PMCSR BER (0xE2)



Bits	Mnemonic	Field Name	Description
D7:D0	PMCSR_BSE	MCSR_BSE	MCSR_BSE Set this bit to “0” since this module is not a PCI-PCI bridge.

Figure 7.2.26 PCI to PCI Bridge Support Extension Register PMCSR BER

7.2.2.27 Data (0xE3)



Bits	Mnemonic	Field Name	Description
D7:D0	Data	Data	Data Fixed to “0” since the Data Select is not supported.

Figure 7.2.27 Data

7.2.3 OpenHCI Registers

This module has in its memory space OpenHCI-compliant registers listed in the following table. These registers can be accessed by specifying the base address in the Memory Base Address Register.

Open HCI registers are accessible at 32 bits.

Table 7.2.2 USB OpenHCI Operational Register Map

Offset	Size (Bytes)	Name
0x00	4	HcRevision
0x04	4	HcControl
0x08	4	HcCommandStatus
0x0C	4	HcInterruptStatus
0x10	4	HcInterruptEnable
0x14	4	HcInterruptDisable
0x18	4	HcHCCA
0x1C	4	HcPeriodCurrentED
0x20	4	HcControlHeadED
0x24	4	HcControlCurrentED
0x28	4	HcBulkHeadED
0x2C	4	HcBulkCurrentED
0x30	4	HcDoneHead
0x34	4	HcFmInterval
0x38	4	HcFmRemining
0x3C	4	HcFmNumber
0x40	4	HcPeriodicStart
0x44	4	HcLSThreshold
0x48	4	HcRhDescriptorA
0x4C	4	HcRhDescriptorB
0x50	4	HcRhStatus
0x54	4	HcRhPortStatus[1]
0x58	4	HcRhPortStatus[2]

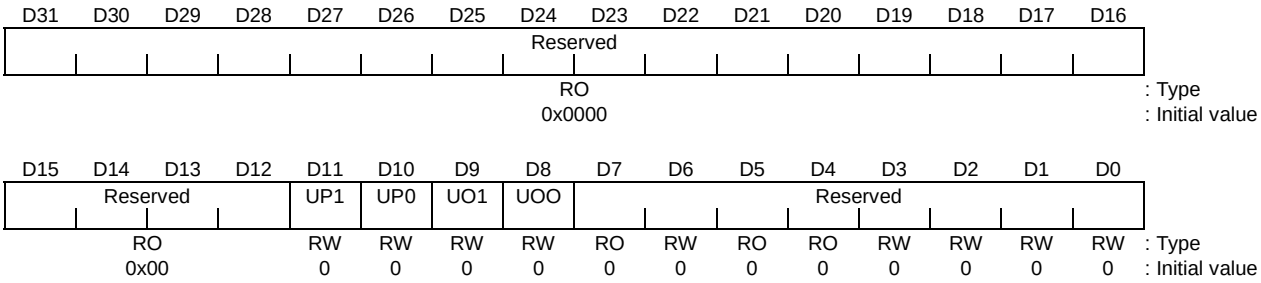
Note: In the Open HCI Specification Release 1.0a, the FrameRemaining (FR) and FrameRemainingToggle (FRT) bits in the HcFmRemaining Register, and the FrameNumber (FN) bits in the HcFmNumber Register are READ ONLY by the Host Controller Driver (HCD). But, this USB 1.1 OHCI Host Controller Core allows these registers to be writeable by the HCD for debugging purposes.
If the HCD write into these registers, undefined results will yield. So please don't write these bits by HCD.

7.2.3.1 Toshiba Original Register Map

Offset	Size (Bytes)	Name
0x100	4	Reserved
0x104	4	TSB_Reserved Register
0x108	4	Reserved

For Test use only.

7.2.3.2 TSB_Reserved Register (0x104)

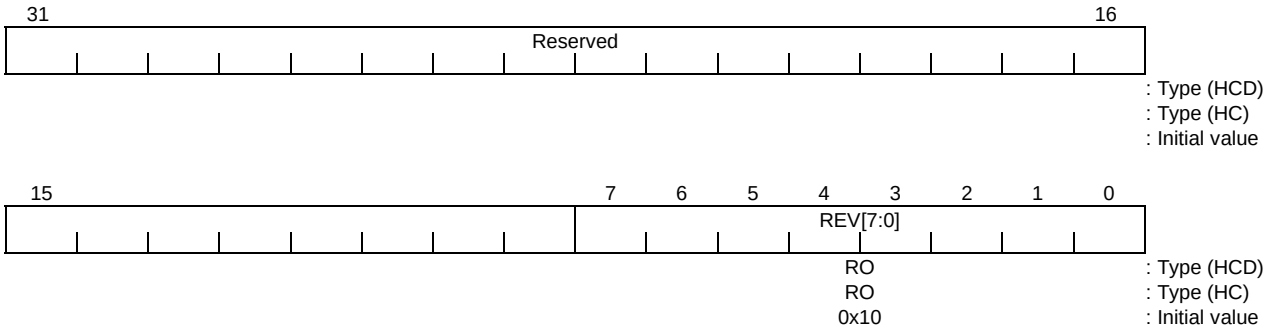


Bits	Mnemonic	Field Name	Description
D31:D12	Reserved		
D11	UP1	UHRPRTPWR phase control	UHRPRTPWR phase control 1: Low=power_on 0: High=power_on
D10	UP0	UHRPRTPWR phase control	UHRPRTPWR phase control 1: Low=power_on 0: High=power_on
D9	UO1	UHPRTCURRENT	UHPRTCURRENT phase control 1: High=over_current 0: Low=over_current
D8	UO0	UHPRTCURRENT	UHPRTCURRENT phase control 1: High=over_current 0: Low=over_current
D7:D0	Reserved		

Figure 7.2.28 TSB_Reserved Register

< The Control and Status Partition >

7.2.3.3 HcRevision Register (HCRR)

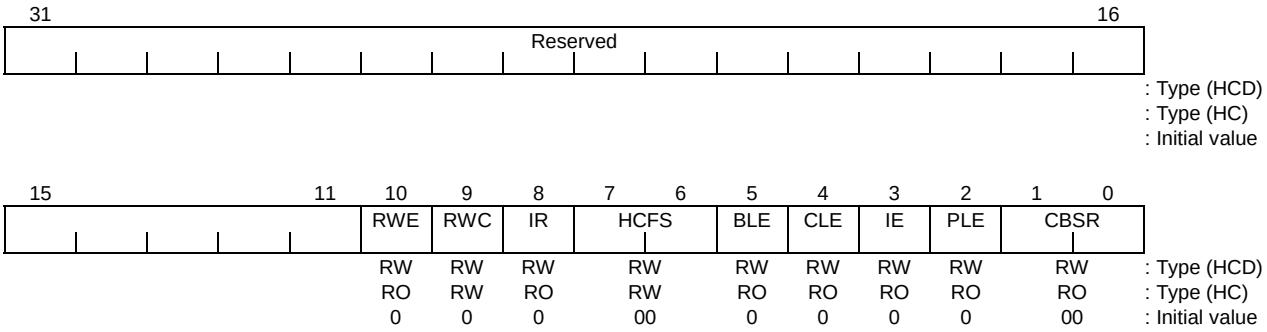


Bits	Mnemonic	Field Name	Description
31:8		Reserved	
7:0	REV	Revision	This read-only field contains the BCD representation of the version of the HCI specification that is implemented by this HC. For example, a value of 0x11 corresponds to version 1.1. All of the HC implementations that are compliant with this specification will have a value of 0x10.

Figure 7.2.29 HcRevision Register

7.2.3.4 HcControl Register

The *HcControl* register defines the operating modes for the Host Controller. Most of the fields in this register are modified only by the Host Controller Driver, except **HostControllerFunctionalState** and **RemoteWakeupConnected**.



Bits	Mnemonic	Field Name	Description
31:11		Reserved	
10	RWE	RemoteWakeup Enable	This bit is used by HCD to enable or disable the remote wakeup feature upon the detection of upstream resume signaling. When this bit is set and the **ResumeDetected** bit in *HcInterruptStatus* is set, a remote wakeup is signaled to the host system. Setting this bit has no impact on the generation of hardware interrupt.
9	RWC	RemoteWakeup Connected	This bit indicates whether HC supports remote wakeup signaling. If remote wakeup is supported and used by the system it is theresponsibility of system firmware to set this bit during POST. HCclears the bit upon a hardware reset but does not alter it upon asoftware reset. Remote wakeup signaling of the host system ishost-bus-specific and is not described in this specification.
8	IR	InterruptRouting	This bit determines the routing of interrupts generated by events registered in *HcInterruptStatus*. If clear, all interrupts are routed to the normal host bus interrupt mechanism. If set, interrupts are routed to the System Management Interrupt. HCD clears this bit upon a hardware reset, but it does not alter this bit upon a software reset. HCD uses this bit as a tag to indicate the ownership of HC.
7:6	HCFS	HostController FunctionalState for USB	00: USBRESET 01: USBRESUME 10: USBOPERATIONAL 11: USBSUSPEND A transition to USBOPERATIONAL from another state causes SOF generation to begin 1 ms later. HCD may determine whether HC has begun sending SOFs by reading the **StartofFrame** field of *HcInterruptStatus*. This field may be changed by HC only when in the USBSUSPEND state. HC may move from the USBSUSPEND state to the USBRESUME state after detecting the resume signaling from a downstream port. HC enters USBSUSPEND after a software reset, whereas it enters USBRESET after a hardware reset. The latter also resets the Root Hub and asserts subsequent reset signaling to downstream ports.

Figure 7.2.30 HcControl Register (1/2)

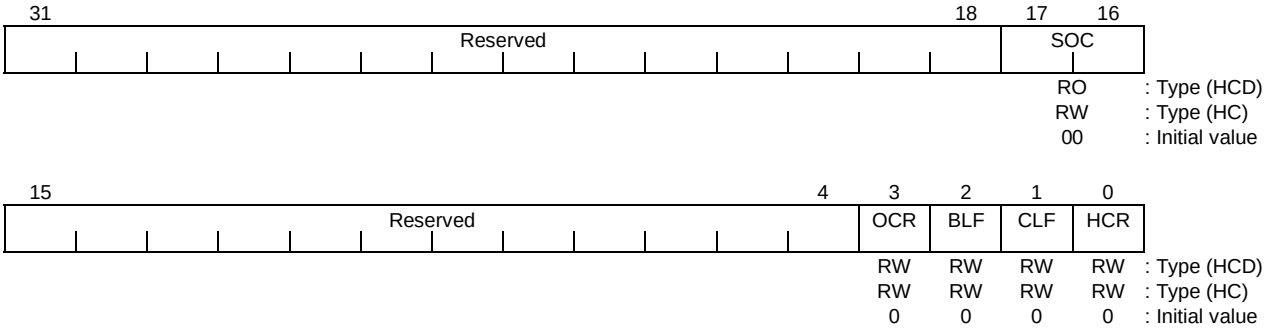
Bits	Mnemonic	Field Name	Description
5	BLE	BulkListEnable	This bit is set to enable the processing of the Bulk list in the next Frame. If cleared by HCD, processing of the Bulk list does not occur after the next SOF. HC checks this bit whenever it determines to process the list. When disabled, HCD may modify the list. If *HcBulkCurrentED* is pointing to an ED to be removed, HCD must advance the pointer by updating *HcBulkCurrentED* before re-enabling processing of the list.
4	CLE	ControlListEnable	This bit is set to enable the processing of the Control list in the next Frame. If cleared by HCD, processing of the Control list does not occur after the next SOF. HC must check this bit whenever it determines to process the list. When disabled, HCD may modify the list. If *HcControlCurrentED* is pointing to an ED to be removed, HCD must advance the pointer by updating *HcControlCurrentED* before re-enabling processing of the list.
3	IE	Isochronous Enable	This bit is used by HCD to enable/disable processing of isochronous EDs. While processing the periodic list in a Frame, HC checks the status of this bit when it finds an Isochronous ED (F=1). If set (enabled), HC continues processing the EDs. If cleared (disabled), HC halts processing of the periodic list (which now contains only isochronous EDs) and begins processing the Bulk/Control lists. Setting this bit is guaranteed to take effect in the next Frame (not the current Frame).
2	PLE	PeriodicList Enable	This bit is set to enable the processing of the periodic list in the next Frame. If cleared by HCD, processing of the periodic list does not occur after the next SOF. HC must check this bit before it starts processing the list.
1:0	CBSR	ControlBulk ServiceRatio	This specifies the service ratio between Control and Bulk EDs. Before processing any of the nonperiodic lists, HC must compare the ratio specified with its internal count on how many nonempty Control EDs have been processed, in determining whether to continue serving another Control ED or switching to Bulk EDs. The internal count will be retained when crossing the frame boundary. In case of reset, HCD is responsible for restoring this value. CBSR No. of Control EDs over Bulk EDs Served 0 1:1 1 2:1 2 3:1 3 4:1

Figure 7.2.30 HcControl Register (2/2)

7.2.3.5 HcCommandStatus Register

The *HcCommandStatus* register is used by the Host Controller to receive commands issued by the Host Controller Driver, as well as reflecting the current status of the Host Controller. To the Host Controller Driver, it appears to be a "write to set" register. The Host Controller must ensure that bits written as '1' become set in the register while bits written as '0' remain unchanged in the register. The Host Controller Driver may issue multiple distinct commands to the Host Controller without concern for corrupting previously issued commands. The Host Controller Driver has normal read access to all bits.

The **SchedulingOverrunCount** field indicates the number of frames with which the Host Controller has detected the scheduling overrun error. This occurs when the Periodic list does not complete before EOF. When a scheduling overrun error is detected, the Host Controller increments the counter and sets the **SchedulingOverrun** field in the *HcInterruptStatus* register.



Bits	Mnemonic	Field Name	Description
31:18		Reserved	
16:17	SOC	Scheduling OverrunCount	These bits are incremented on each scheduling overrun error. It is initialized to 00b and wraps around at 11b. This will be incremented when a scheduling overrun is detected even if **SchedulingOverrun** in *HcInterruptStatus* has already been set. This is used by HCD to monitor any persistent scheduling problems.
15:4		Reserved	
3	OCR	Ownership ChangeRequest	This bit is set by an OS HCD to request a change of control of the HC. When set HC will set the **OwnershipChange** field in *HcInterruptStatus*. After the changeover, this bit is cleared and remains so until the next request from OS HCD.
2	BLF	BulkListFilled	This bit is used to indicate whether there are any TDs on the Bulk list. It is set by HCD whenever it adds a TD to an ED in the Bulk list. When HC begins to process the head of the Bulk list, it checks BF. As long as **BulkListFilled** is 0, HC will not start processing the Bulk list. If **BulkListFilled** is 1, HC will start processing the Bulk list and will set BF to 0. If HC finds a TD on the list, then HC will set **BulkListFilled** to 1 causing the Bulk list processing to continue. If no TD is found on the Bulk list, and if HCD does not set **BulkListFilled**, then **BulkListFilled** will still be 0 when HC completes processing the Bulk list and Bulk list processing will stop.

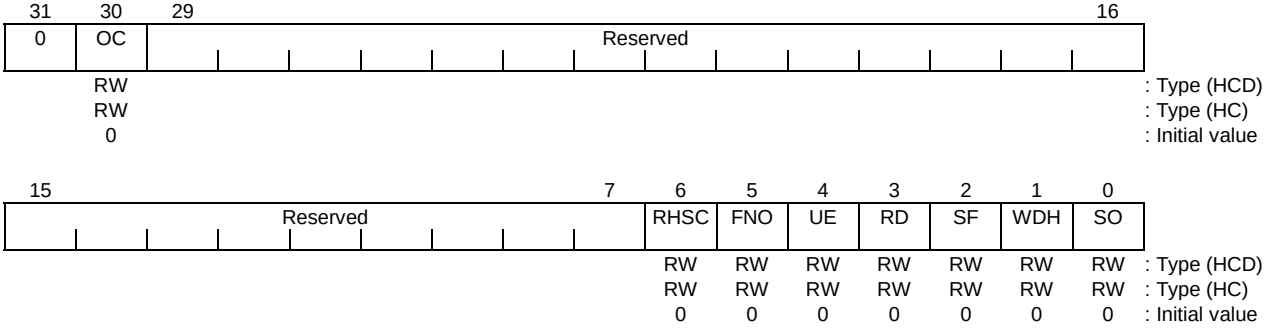
Figure 7.2.31 HcCommandStatus Register (1/2)

Bits	Mnemonic	Field Name	Description
1	CLF	ControlListFilled	This bit is used to indicate whether there are any TDs on the Control list. It is set by HCD whenever it adds a TD to an ED in the Control list. When HC begins to process the head of the Control list, it checks CLF. As long as **ControlListFilled** is 0, HC will not start processing the Control list. If CF is 1, HC will start processing the Control list and will set **ControlListFilled** to 0. If HC finds a TD on the list, then HC will set **ControlListFilled** to 1 causing the Control list processing to continue. If no TD is found on the Control list, and if the HCD does not set **ControlListFilled**, then **ControlListFilled** will still be 0 when HC completes processing the Control list and Control list processing will stop.
0	HCR	HostController Reset	This bit is set by HCD to initiate a software reset of HC. Regardless of the functional state of HC, it moves to the USBSUSPEND state in which most of the operational registers are reset except those stated otherwise; e.g., the **InterruptRouting** field of *HcControl*, and no Host bus accesses are allowed. This bit is cleared by HC upon the completion of the reset operation.The reset operation must be completed within 10 μs. This bit,when set, should not cause a reset to the Root Hub and no subsequent reset signaling should be asserted to its downstream ports.

Figure 7.2.31 HcCommandStatus Register (2/2)

7.2.3.6 HcInterruptStatus Register

This register provides status on various events that cause hardware interrupts. When an event occurs, Host Controller sets the corresponding bit in this register. When a bit becomes set, a hardware interrupt is generated if the interrupt is enabled in the *HcInterruptEnable* register (see Section 7.2.3.7) and the **MasterInterruptEnable** bit is set. The Host Controller Driver may clear specific bits in this register by writing '1' to bit positions to be cleared. The Host Controller Driver may not set any of these bits. The Host Controller will never clear the bit.



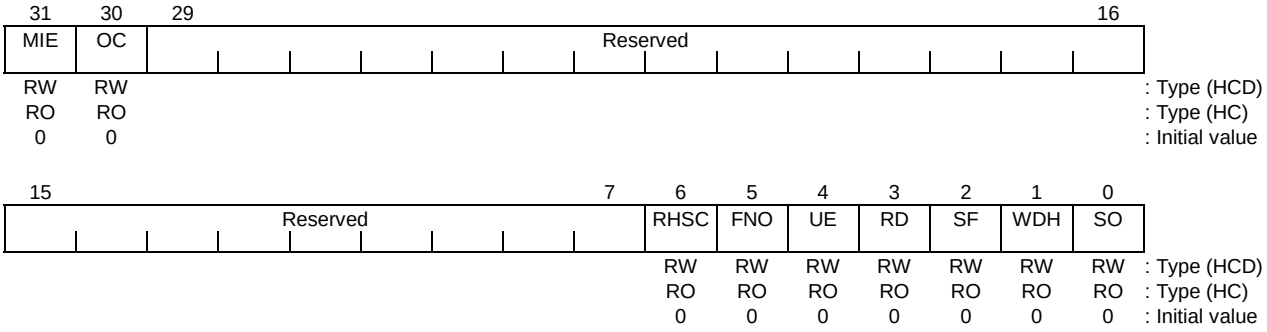
Bits	Mnemonic	Field Name	Description
31		Reserved	
30	OC	Ownership Change	This bit is set by HC when HCD sets the **OwnershipChangeRequest** field in *HcCommandStatus*. This event, when unmasked, will always generate an System Management Interrupt (SMI) immediately. This bit is tied to 0b when the SMI pin is not implemented.
29:7		Reserved	
6	RHSC	RootHubStatus Change	This bit is set when the content of *HcRhStatus* or the content of any of *HcRhPortStatus*[**NumberofDownstreamPort**] has changed.
5	FNO	FrameNumber Overflow	This bit is set when the MSb of *HcFmNumber* (bit 15) changes value, from 0 to 1 or from 1 to 0, and after *HccaFrameNumber* has been updated.
4	UE	Unrecoverable Error	This bit is set when HC detects a system error not related to USB. HC should not proceed with any processing nor signaling before the system error has been corrected. HCD clears this bit after HC has been reset.
3	RD	ResumeDetected	This bit is set when HC detects that a device on the USB is asserting resume signaling. It is the transition from no resume signaling to resume signaling causing this bit to be set. This bit is not set when HCD sets the USBRESUME state.
2	SF	StartofFrame	This bit is set by HC at each start of a frame and after the update of *HccaFrameNumber*. HC also generates a SOF token at the same time.
1	WDH	WritebackDone Head	This bit is set immediately after HC has written *HcDoneHead* to *HccaDoneHead*. Further updates of the *HccaDoneHead* will not occur until this bit has been cleared. HCD should only clear this bit after it has saved the content of *HccaDoneHead*.
0	SO	Scheduling Overrun	This bit is set when the USB schedule for the current Frame overruns and after the update of *HccaFrameNumber*. A scheduling overrun will also cause the **SchedulingOverrunCount** of *HcCommandStatus* to be incremented.

Figure 7.2.32 HcInterruptStatus Register

7.2.3.7 HcInterruptEnable Register

Each enable bit in the *HcInterruptEnable* register corresponds to an associated interrupt bit in the *HcInterruptStatus* register. The *HcInterruptEnable* register is used to control which events generate a hardware interrupt. When a bit is set in the *HcInterruptStatus* register AND the corresponding bit in the *HcInterruptEnable* register is set AND the **MasterInterruptEnable** bit is set, then a hardware interrupt is requested on the host bus.

Writing a '1' to a bit in this register sets the corresponding bit, whereas writing a '0' to a bit in this register leaves the corresponding bit unchanged. On read, the current value of this register is returned.

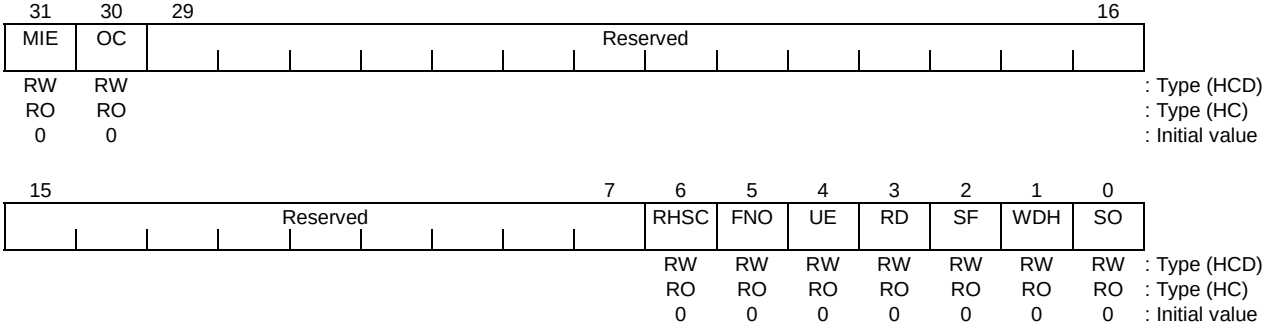


Bits	Mnemonic	Field Name	Description
31	MIE	MasterInterrupt Enable	A '0' written to this field is ignored by HC. A '1' written to this field enables interrupt generation due to events specified in the other bits of this register. This is used by HCD as a Master Interrupt Enable.
30	OC	Ownership Change	0 - Ignore 1 - Enable interrupt generation due to Ownership Change.
29:7		Reserved	
6	RHSC	RootHubStatus Change	0 - Ignore 1 - Enable interrupt generation due to Root Hub Status Change.
5	FNO	FrameNumber Overflow	0 - Ignore 1 - Enable interrupt generation due to Frame Number Overflow.
4	UE	Unrecoverable Error	0 - Ignore 1 - Enable interrupt generation due to Unrecoverable Error.
3	RD	ResumeDetected	0 - Ignore 1 - Enable interrupt generation due to Resume Detect.
2	SF	StartofFrame	0 - Ignore 1 - Enable interrupt generation due to Start of Frame.
1	WDH	WritebackDone Head	0 - Ignore 1 - Enable interrupt generation due to HcDoneHead Writeback.
0	SO	Scheduling Overrun	0 - Ignore 1 - Enable interrupt generation due to Scheduling Overrun.

Figure 7.2.33 HcInterruptEnable Register

7.2.3.8 HcInterruptDisable Register

Each disable bit in the *HcInterruptDisable* register corresponds to an associated interrupt bit in the *HcInterruptStatus* register. The *HcInterruptDisable* register is coupled with the *HcInterruptEnable* register. Thus, writing a '1' to a bit in this register clears the corresponding bit in the *HcInterruptEnable* register, whereas writing a '0' to a bit in this register leaves the corresponding bit in the *HcInterruptEnable* register unchanged. On read, the current value of the *HcInterruptEnable* register is returned.



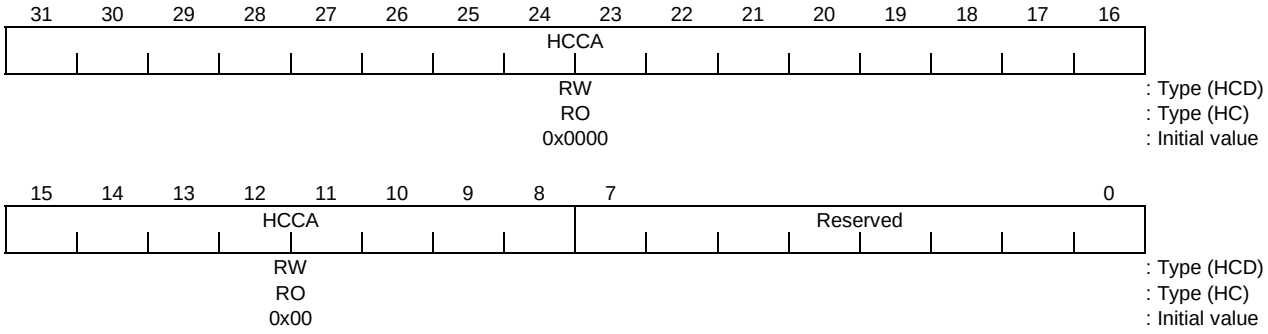
Bits	Mnemonic	Field Name	Description
31	MIE	MasterInterrupt Enable	A '0' written to this field is ignored by HC. A '1' written to this field disables interrupt generation due to events specified in the other bits of this register. This field is set after a hardware or software reset.
30	OC	Ownership Change	0 - Ignore 1 - Disable interrupt generation due to Ownership Change.
29:7		Reserved	
6	RHSC	RootHubStatus Change	0 - Ignore 1 - Disable interrupt generation due to Root Hub Status Change.
5	FNO	FrameNumber Overflow	0 - Ignore 1 - Disable interrupt generation due to Frame Number Overflow.
4	UE	Unrecoverable Error	0 - Ignore 1 - Disable interrupt generation due to Unrecoverable Error.
3	RD	ResumeDetected	0 - Ignore 1 - Disable interrupt generation due to Resume Detect.
2	SF	StartofFrame	0 - Ignore 1 - Disable interrupt generation due to Start of Frame.
1	WDH	WritebackDone Head	0 - Ignore 1 - Disable interrupt generation due to HcDoneHead Writeback.
0	SO	Scheduling Overrun	0 - Ignore 1 - Disable interrupt generation due to Scheduling Overrun.

Figure 7.2.34 HcInterruptDisable Register

< Memory Pointer Partition >

7.2.3.9 HcHCCA Register

The *HcHCCA* register contains the physical address of the Host Controller Communication Area. The Host Controller Driver determines the alignment restrictions by writing all 1s to *HcHCCA* and reading the content of *HcHCCA*. The alignment is evaluated by examining the number of zeroes in the lower order bits. The minimum alignment is 256 bytes; therefore, bits 0 through 7 must always return '0' when read. This area is used to hold the control structures and the Interrupt table that are accessed by both the Host Controller and the Host Controller Driver.

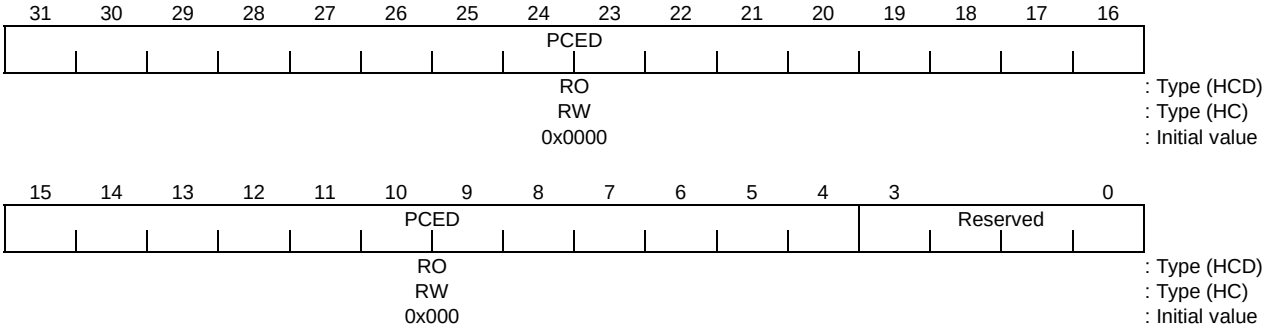


Bits	Mnemonic	Field Name	Description
31:8	HCCA	Host Controller Communication Area	This is the base address of the Host Controller Communication Area.
7:0		Reserved	

Figure 7.2.35 HcHCCA Register

7.2.3.10 HcPeriodCurrentED Register

The *HcPeriodCurrentED* register contains the physical address of the current Isochronous or Interrupt Endpoint Descriptor.

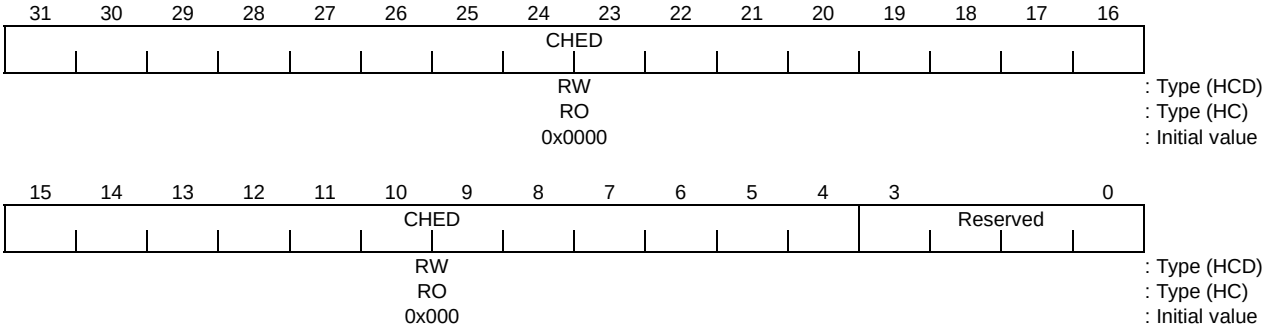


Bits	Mnemonic	Field Name	Description
31:4	PCED	PeriodCurrentED	This is used by HC to point to the head of one of the Periodic lists which will be processed in the current Frame. The content of this register is updated by HC after a periodic ED has been processed. HCD may read the content in determining which ED is currently being processed at the time of reading.
3:0		Reserved	

Figure 7.2.36 HcPeriodCurrentED Register

7.2.3.11 HcControlHeadED Register

The *HcControlHeadED* register contains the physical address of the first Endpoint Descriptor of the Control list.

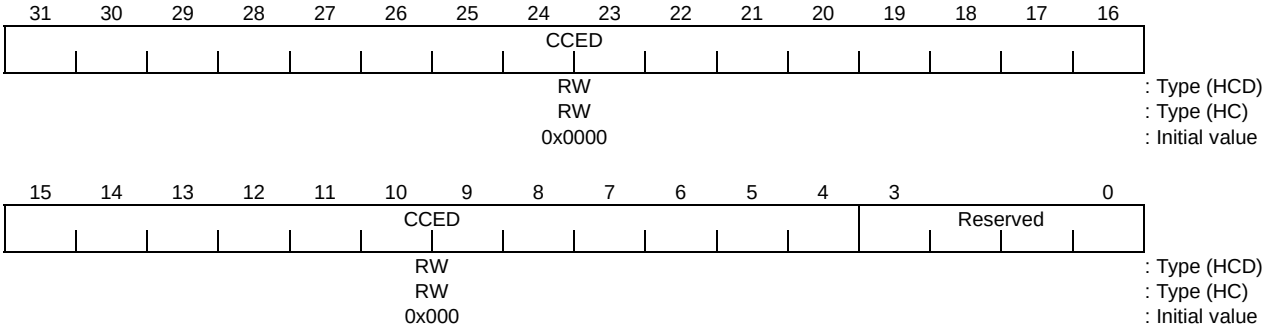


Bits	Mnemonic	Field Name	Description
31:4	CHED	ControlHeadED	HC traverses the Control list starting with the *HcControlHeadED* pointer. The content is loaded from HCCA during the initialization of HC.
3:0		Reserved	

Figure 7.2.37 HcControlHeadED Register

7.2.3.12 HcControlCurrentED Register

The *HcControlCurrentED* register contains the physical address of the current Endpoint Descriptor of the Control list.

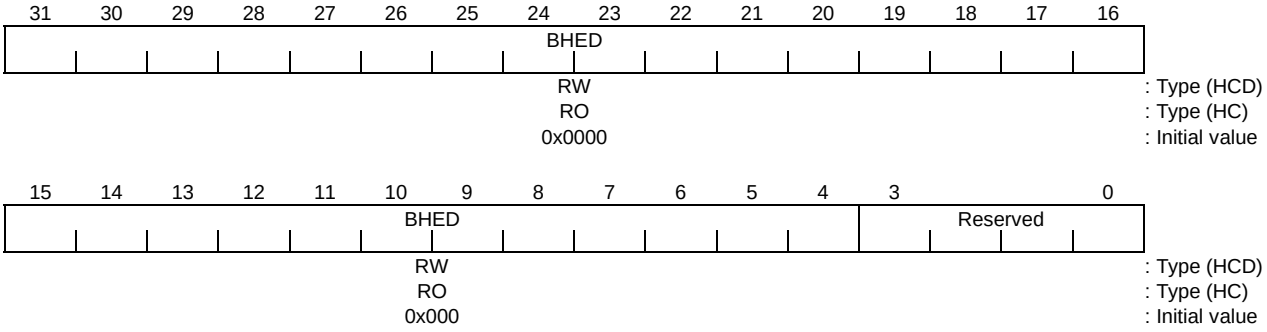


Bits	Mnemonic	Field Name	Description
31:4	CCED	ControlCurrentED	This pointer is advanced to the next ED after serving the present one. HC will continue processing the list from where it left off in the last Frame. When it reaches the end of the Control list, HC checks the **ControlListFilled** of in *HcCommandStatus*. If set, it copies the content of *HcControlHeadED* to *HcControlCurrentED* and clears the bit. If not set, it does nothing. HCD is allowed to modify this register only when the **ControlListEnable** of *HcControl* is cleared. When set, HCD only reads the instantaneous value of this register. Initially, this is set to zero to indicate the end of the Control list.
3:0		Reserved	

Figure 7.2.38 HcControlCurrentED Register

7.2.3.13 HcBulkHeadED Register

The *HcBulkHeadED* register contains the physical address of the first Endpoint Descriptor of the Bulk list.

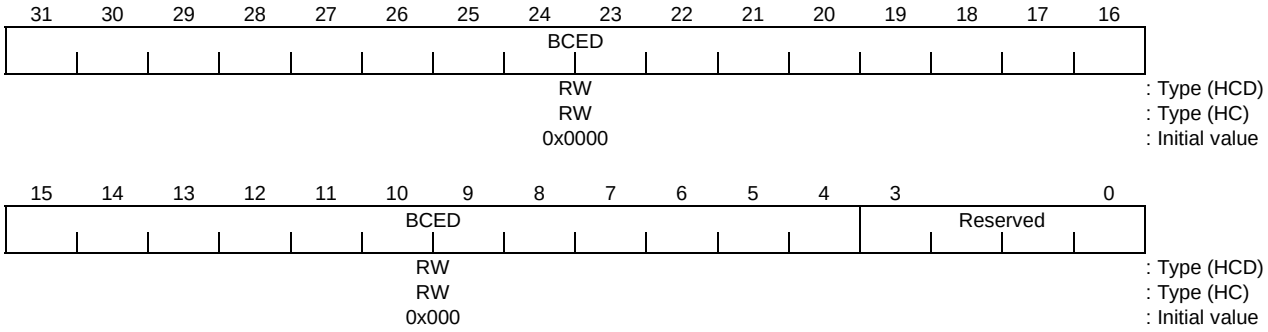


Bits	Mnemonic	Field Name	Description
31:4	BHED	BulkHeadED	HC traverses the Bulk list starting with the *HcBulkHeadED* pointer. The content is loaded from HCCA during the initialization of HC.
3:0		Reserved	

Figure 7.2.39 HcBulkHeadED Register

7.2.3.14 HcBulkCurrentED Register

The *HcBulkCurrentED* register contains the physical address of the current endpoint of the Bulk list. As the Bulk list will be served in a round-robin fashion, the endpoints will be ordered according to their insertion to the list.

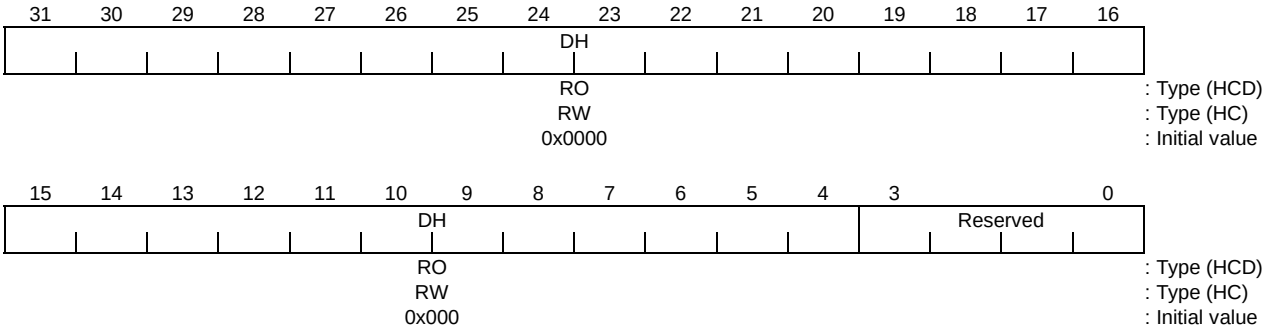


Bits	Mnemonic	Field Name	Description
31:4	BCED	BulkCurrentED	This is advanced to the next ED after the HC has served the present one. HC continues processing the list from where it left off in the last Frame. When it reaches the end of the Bulk list, HC checks the **ControlListFilled** of HcControl. If set, it copies the content of *HcBulkHeadED* to *HcBulkCurrentED* and clears the bit. If it is not set, it does nothing. HCD is only allowed to modify this register when the **BulkListEnable** of *HcControl* is cleared. When set, the HCD only reads the instantaneous value of this register. This is initially set to zero to indicate the end of the Bulk list.
3:0		Reserved	

Figure 7.2.40 HcBulkCurrentED Register

7.2.3.15 HcDoneHead Register

The *HcDoneHead* register contains the physical address of the last completed Transfer Descriptor that was added to the Done queue. In normal operation, the Host Controller Driver should not need to read this register as its content is periodically written to the HCCA.



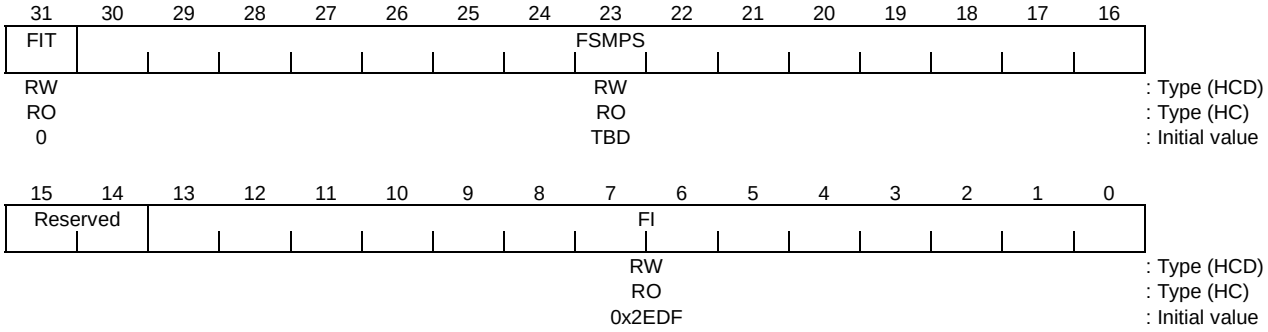
Bits	Mnemonic	Field Name	Description
31:4	DH	DoneHead	When a TD is completed, HC writes the content of *HcDoneHead* to the NextTD field of the TD. HC then overwrites the content of *HcDoneHead* with the address of this TD. This is set to zero whenever HC writes the content of this register to HCCA. It also sets the **WritebackDoneHead** of *HcInterruptStatus*.
3:0		Reserved	

Figure 7.2.41 HcDoneHead Register

< Frame Counter Partition >

7.2.3.16 HcFmIteraval Register

The *HcFmInterval* register contains a 14-bit value which indicates the bit time interval in a Frame, (i.e., between two consecutive SOFs), and a 15-bit value indicating the Full Speed maximum packet size that the Host Controller may transmit or receive without causing scheduling overrun. The Host Controller Driver may carry out minor adjustment on the **FrameInterval** by writing a new value over the present one at each SOF. This provides the programmability necessary for the Host Controller to synchronize with an external clocking resource and to adjust any unknown local clock offset.

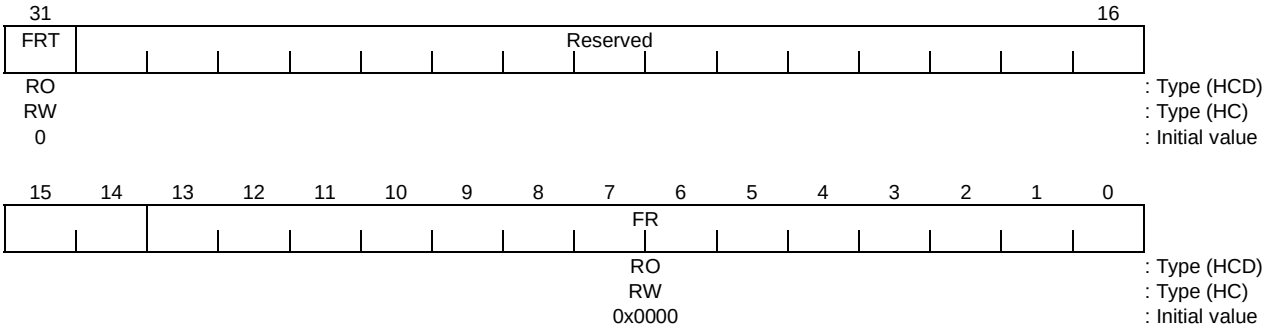


Bits	Mnemonic	Field Name	Description
31	FIT	FrameInterval Toggle	HCD toggles this bit whenever it loads a new value to **FrameInterval**.
30:16	FSMPS	FSLargestData Packet	This field specifies a value which is loaded into the Largest Data Packet Counter at the beginning of each frame. The counter value represents the largest amount of data in bits which can be sent or received by the HC in a single transaction at any given time without causing scheduling overrun. The field value is calculated by the HCD.
15:14		Reserved	
13:0	FI	FrameInterval	This specifies the interval between two consecutive SOFs in bit times. The nominal value is set to be 11,999. HCD should store the current value of this field before resetting HC. By setting the **HostControllerReset** field of *HcCommandStatus* as this will cause the HC to reset this field to its nominal value. HCD may choose to restore the stored value upon the completion of the Reset sequence.

Figure 7.2.42 HcFmIterval Register

7.2.3.17 HcFmRemaining Register

The *HcFmRemaining* register is a 14-bit down counter showing the bit time remaining in the current Frame.

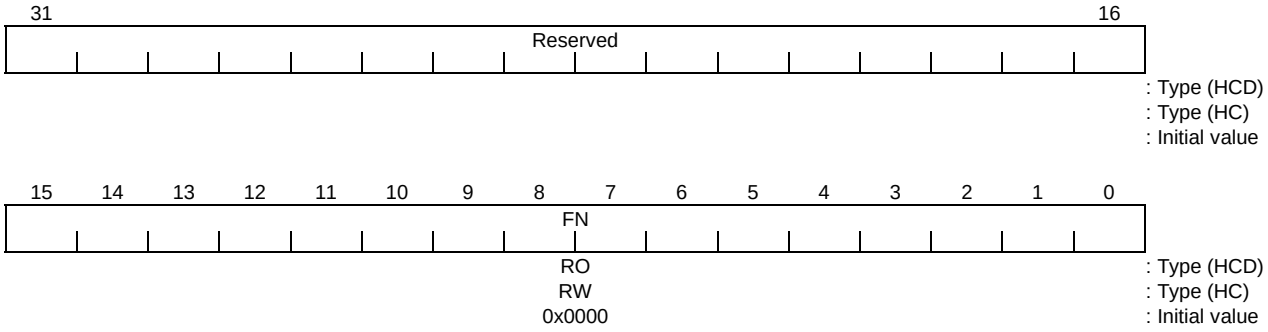


Bits	Mnemonic	Field Name	Description
31	FRT	FrameRemaining Toggle	This bit is loaded from the **FrameIntervalToggle** field of *HcFmInterval* whenever **FrameRemaining** reaches 0. This bit is used by HCD for the synchronization between **FrameInterval** and **FrameRemaining**.
30:14		Reserved	
13:0	FR	FrameRemaining	This counter is decremented at each bit time. When it reaches zero, it is reset by loading the **FrameInterval** value specified in *HcFmInterval* at the next bit time boundary. When entering the USBOPERATIONAL state, HC re-loads the content with the **FrameInterval** of *HcFmInterval* and uses the updated value from the next SOF.

Figure 7.2.43 HcFmRemaining Register

7.2.3.18 HcFmNumber Register

The *HcFmNumber* register is a 16-bit counter. It provides a timing reference among events happening in the Host Controller and the Host Controller Driver. The Host Controller Driver may use the 16-bit value specified in this register and generate a 32-bit frame number without requiring frequent access to the register.

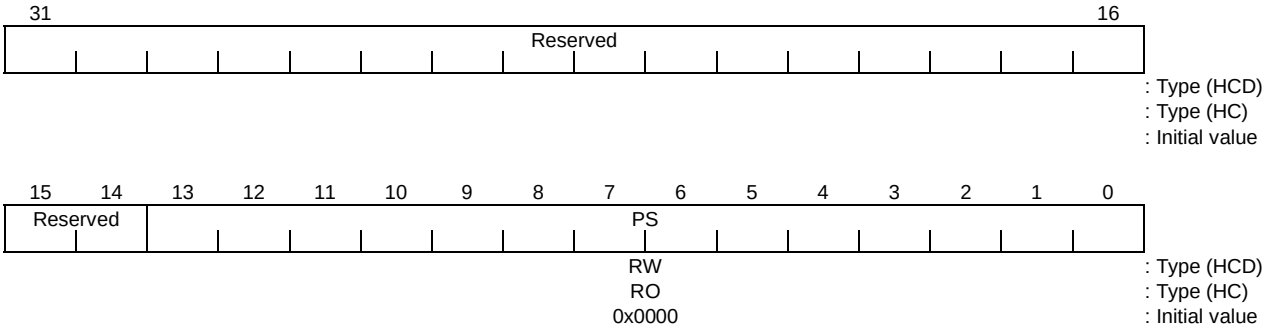


Bits	Mnemonic	Field Name	Description
31:16		Reserved	
15:0	FN	FrameNumber	This is incremented when *HcFmRemaining* is re-loaded. It will be rolled over to 0x00 after 0xffff. When entering the USBOPERATIONAL state, this will be incremented automatically. The content will be written to HCCA after HC has incremented the **FrameNumber** at each frame boundary and sent a SOF but before HC reads the first ED in that Frame. After writing to HCCA, HC will set the **StartofFrame** in *HcInterruptStatus.*

Figure 7.2.44 HcFmNumber Register

7.2.3.19 HcPeriodicStart Register

The *HcPeriodicStart* register has a 14-bit programmable value which determines when is the earliest time HC should start processing the periodic list.

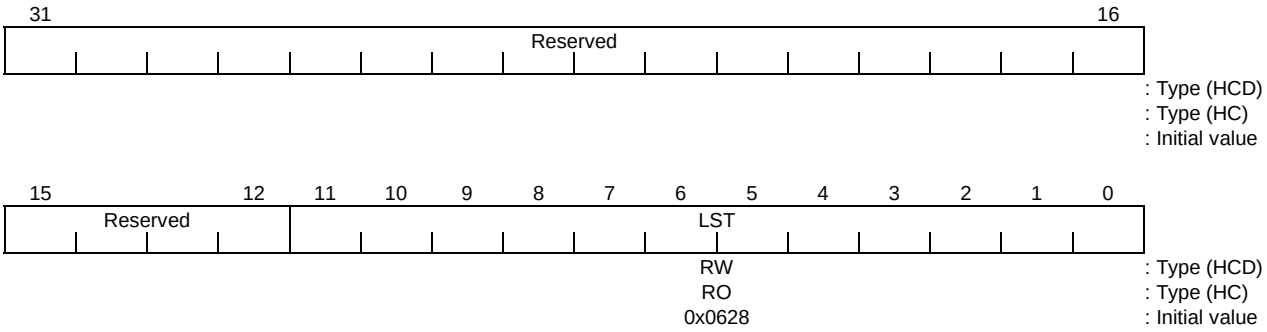


Bits	Mnemonic	Field Name	Description
31:14		Reserved	
13:0	PS	PeriodicStart	After a hardware reset, this field is cleared. This is then set by HCD during the HC initialization. The value is calculated roughly as 10% off from *HcFmInterval*.. A typical value will be 0x3E67. When *HcFmRemaining* reaches the value specified, processing of the periodic lists will have priority over Control/Bulk processing. HC will therefore start processing the Interrupt list after completing the current Control or Bulk transaction that is in progress.

Figure 7.2.45 HcPeriodicStart Register

7.2.3.20 HcLSThreshold Register

The *HcLSThreshold* register contains an 11-bit value used by the Host Controller to determine whether to commit to the transfer of a maximum of 8-byte LS packet before EOF. Neither the Host Controller nor the Host Controller Driver is allowed to change this value.



Bits	Mnemonic	Field Name	Description
31:12		Reserved	
11:0	LST	LSThreshold	This field contains a value which is compared to the **FrameRemaining** field prior to initiating a Low Speed transaction. The transaction is started only if **FrameRemaining** ≥ this field. The value is calculated by HCD with the consideration of transmission and setup overhead.

Figure 7.2.46 HcLSThreshold Register

< Root Hub Partition >

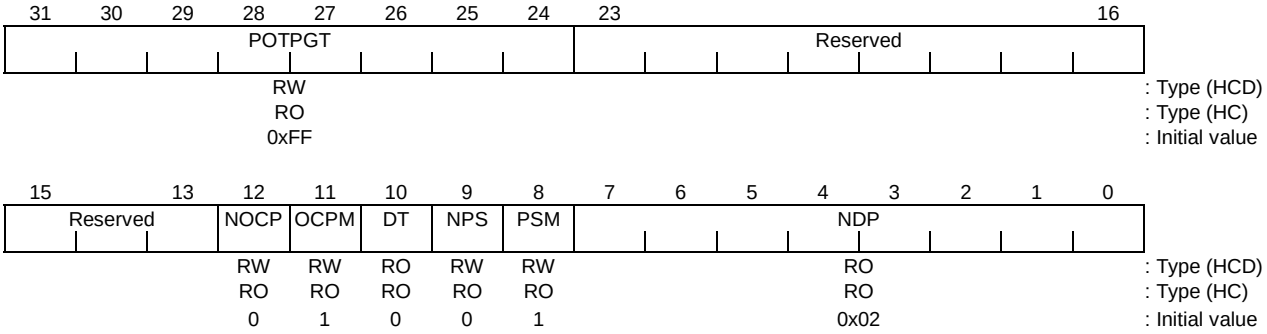
All registers included in this partition are dedicated to the USB Root Hub which is an integral part of the Host Controller though still a functionally separate entity. The HCD emulates USBD accesses to the Root Hub via a register interface. The HCD maintains many USB-defined hub features which are not required to be supported in hardware. For example, the Hub's Device, Configuration, Interface, and Endpoint Descriptors are maintained only in the HCD as well as some static fields of the Class Descriptor. The HCD also maintains and decodes the Root Hub's device address as well as other trivial operations which are better suited to software than hardware.

The Root Hub register interface is otherwise developed to maintain similarity of bit organization and operation to typical hubs which are found in the system. Below are four register definitions: *HcRhDescriptorA*, *HcRhDescriptorB*, *HcRhStatus*, and *HcRhPortStatus[1:NDP]*. Each register is read and written as a Dword. These registers are only written during initialization to correspond with the system implementation. The *HcRhDescriptorA* and *HcRhDescriptorB* registers should be implemented such that they are writeable regardless of the HC USB state. *HcRhStatus* and *HcRhPortStatus* must be writeable during the USBOPERATIONAL state.

Note: IS denotes an implementation-specific reset value for that field.

7.2.3.21 HcRhDescriptorA Register

The *HcRhDescriptorA* register is the first register of two describing the characteristics of the Root Hub. Reset values are implementation-specific. The descriptor length, descriptor type, and hub controller current fields of the hub Class Descriptor are emulated by the HCD. All other fields are located in the *HcRhDescriptorA* and *HcRhDescriptorB* registers.



Bits	Mnemonic	Field Name	Description
31:24	POTPGT	PowerOnTo PowerGoodTime	This byte specifies the duration HCD has to wait before accessing a powered-on port of the Root Hub. It is implementation-specific. The unit of time is 2 ms. The duration is calculated as **POTPGT** * 2 ms.
23:13		Reserved	
12	NOCP	NoOverCurrent Protection	This bit describes how the overcurrent status for the Root Hub ports are reported. When this bit is cleared, the **OverCurrentProtectionMode** field specifies global or per-port reporting. 0: Over-current status is reported collectively for all downstream ports 1: No overcurrent protection supported
11	OCPM	OverCurrent ProtectionMode	This bit describes how the overcurrent status for the Root Hub ports are reported. At reset, this fields should reflect the same mode as **PowerSwitchingMode**. This field is valid only if the **NoOverCurrentProtection** field is cleared. 0: Over-current status is reported collectively for all downstream ports 1: Over-current status is reported on a per-port basis
10	DT	DeviceType	This bit specifies that the Root Hub is not a compound device. The Root Hub is not permitted to be a compound device. This field should always read/write 0.
9	NPS	NoPower Switching	These bits are used to specify whether power switching is supported or ports are always powered. It is implementation-specific. When this bit is cleared, the **PowerSwitchingMode** specifies global or per-port switching. 0: Ports are power switched 1: Ports are always powered on when the HC is powered on

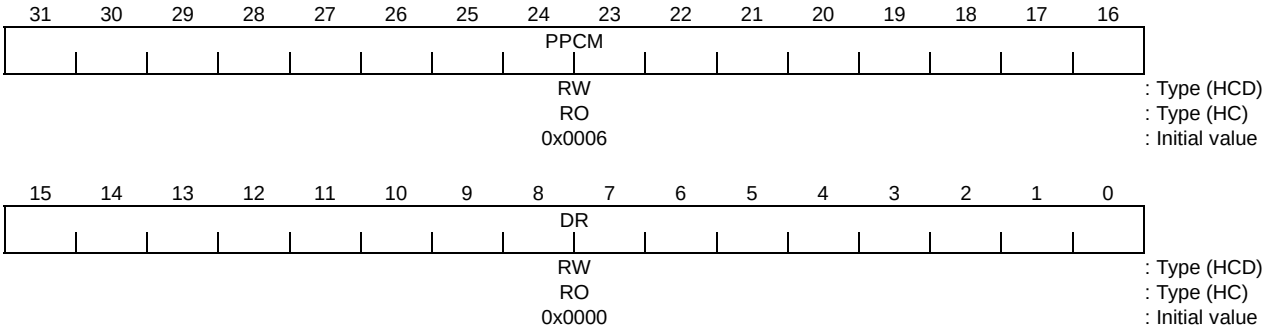
Figure 7.2.47 HcRhDescriptorA Register (1/2)

Bits	Mnemonic	Field Name	Description
8	PSM	PowerSwitching Mode	This bit is used to specify how the power switching of the Root Hub ports is controlled. It is implementation-specific. This field is only valid if the **NoPowerSwitching** field is cleared. 0: All ports are powered at the same time. 1: Each port is powered individually. This mode allows port power to be controlled by either the global switch or per-port switching. If the **PortPowerControlMask** bit is set, the port responds only to port power commands (**Set/ClearPortPower**). If the port mask is cleared, then the port is controlled only by the global power switch (**Set/ClearGlobalPower**).
7:0	NDP	Number DownstreamPorts	These bits specify the number of downstream ports supported by the Root Hub. It is implementation-specific. Set NDP to “0x02” since the number of ports of this module is 2.

Figure 7.2.47 HcRhDescriptorA Register (2/2)

7.2.3.22 HcRhDescriptorB Register

The *HcRhDescriptorB* register is the second register of two describing the characteristics of the Root Hub. These fields are written during initialization to correspond with the system implementation. Reset values are implementation-specific.

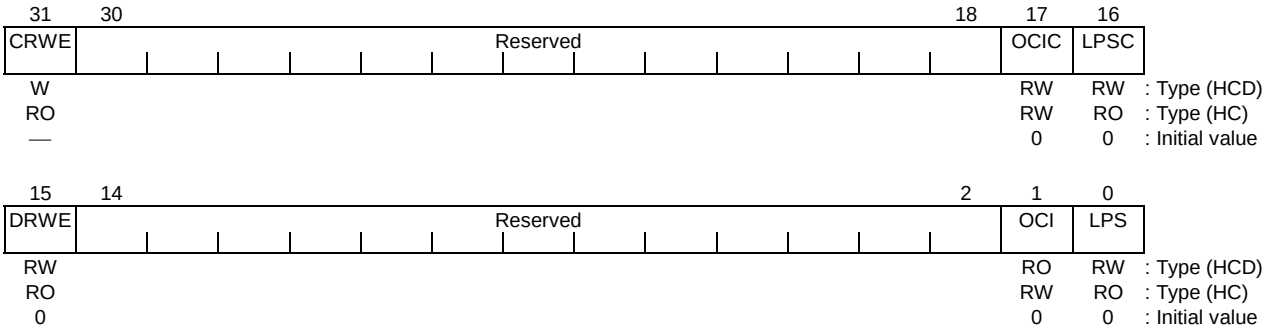


Bits	Mnemonic	Field Name	Description
31:16	PPCM	PortPowerControl Mask	Each bit indicates if a port is affected by a global power control command when **PowerSwitchingMode** is set. When set, the port's power state is only affected by per-port power control (**Set/ClearPortPower**). When cleared, the port is controlled by the global power switch (**Set/ClearGlobalPower**). If the device is configured to global switching mode (**PowerSwitchingMode**=0), this field is not valid. bit 0: Reserved bit 1: Ganged-power mask on Port #1 bit 2: Ganged-power mask on Port #2 ... bit 15: Ganged-power mask on Port #15
15:0	DR	Device Removable	Each bit is dedicated to a port of the Root Hub. When cleared, the attached device is removable. When set, the attached device is not removable. bit 0: Reserved bit 1: Device attached to Port #1 bit 2: Device attached to Port #2 ... bit 15: Device attached to Port #15

Figure 7.2.48 HcRhDescriptorB Register

7.2.3.23 HcRhStatus Register

The *HcRhStatus* register is divided into two parts. The lower word of a Dword represents the **Hub Status** field and the upper word represents the **Hub Status Change** field. Reserved bits should always be written '0'.



Bits	Mnemonic	Field Name	Description
31	CRWE	ClearRemote WakeupEnable	(write) **ClearRemoteWakeupEnable** Writing a '1' clears **DeviceRemoveWakeupEnable**. Writing a '0' has no effect.
30:18		Reserved	
17	OCIC	Over CurrentIndicator Change	This bit is set by hardware when a change has occurred to the OCI field of this register. The HCD clears this bit by writing a '1'. Writing a '0' has no effect.
16	LPSC	LocalPower StatusChange	(read) **LocalPowerStatusChange** The Root Hub does not support the local power status feature; thus, this bit is always read as '0'. (write) **SetGlobalPower** In global power mode (**PowerSwitchingMode**=0), This bit is written to '1' to turn on power to all ports (clear **PortPowerStatus**). In per-port power mode, it sets **PortPowerStatus** only on ports whose **PortPowerControlMask** bit is not set. Writing a '0' has no effect.
15	DRWE	DeviceRemote WakeupEnable	(read) **DeviceRemoteWakeupEnable** This bit enables a **ConnectStatusChange** bit as a resume event, causing a USBSUSPEND to USBRESUME state transition and setting the **ResumeDetected** interrupt. 0 = **ConnectStatusChange** is not a remote wakeup event. 1 = **ConnectStatusChange** is a remote wakeup event. (write) **SetRemoteWakeupEnable** Writing a '1' sets **DeviceRemoveWakeupEnable**. Writing a '0' has no effect.
14:2		Reserved	
1	OCI	Over CurrentIndicator	This bit reports overcurrent conditions when the global reporting is implemented. When set, an overcurrent condition exists. When cleared, all power operations are normal. If per-port overcurrent protection is implemented this bit is always '0'

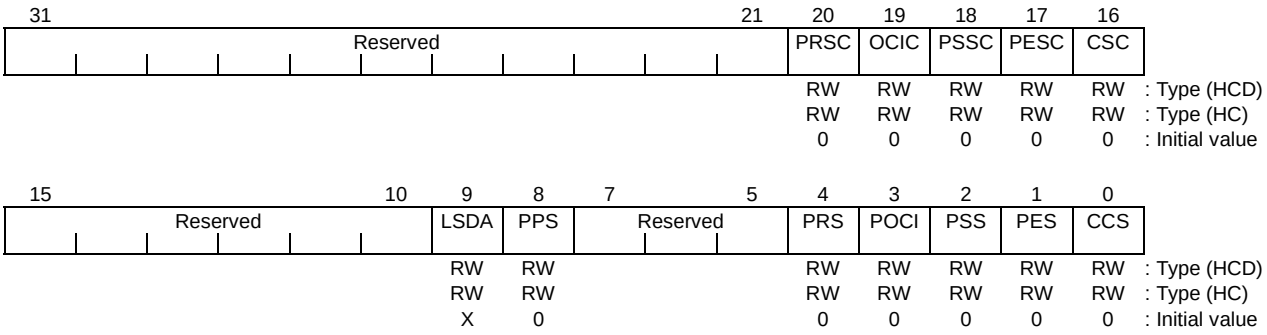
Figure 7.2.49 HcRhStatus Register (1/2)

Bits	Mnemonic	Field Name	Description
0	LPS	LocalPower Status	(read) **LocalPowerStatus** The Root Hub does not support the local power status feature; thus, this bit is always read as '0'. (write) **ClearGlobalPower** In global power mode (**PowerSwitchingMode**=0), This bit is written to '1' to turn off power to all ports (clear **PortPowerStatus**). In per-port power mode, it clears **PortPowerStatus** only on ports whose **PortPowerControlMask** bit is not set. Writing a '0' has no effect.

Figure 7.2.49 HcRhStatus Register (2/2)

7.2.3.24 HcRhPortStatus[1:2] Register

The *HcRhPortStatus*[1:2] register is used to control and report port events on a per-port basis. **NumberDownstreamPorts** represents the number of *HcRhPortStatus* registers that are implemented in hardware. The lower word is used to reflect the port status, whereas the upper word reflects the status change bits. Some status bits are implemented with special write behavior (see below). If a transaction (token through handshake) is in progress when a write to change port status occurs, the resulting port status change must be postponed until the transaction completes. Reserved bits should always be written '0'.



Bits	Mnemonic	Field Name	Description
31:21		Reserved	
20	PRSC	PortResetStatus Change	This bit is set at the end of the 10-ms port reset signal. The HCD writes a '1' to clear this bit. Writing a '0' has no effect. 0 = Port reset is not complete 1 = Port reset is complete
19	COIC	PortOver CurrentIndicator Change	This bit is valid only if overcurrent conditions are reported on a per-port basis. This bit is set when Root Hub changes the **PortOverCurrentIndicator** bit. The HCD writes a '1' to clear this bit. Writing a '0' has no effect. 0 = No change in **PortOverCurrentIndicator** 1 = **PortOverCurrentIndicator** has changed
18	PSSC	PortSuspend StatusChange	This bit is set when the full resume sequence has been completed. This sequence includes the 20-s resume pulse, LS EOP, and 3-ms resynchronization delay. The HCD writes a '1' to clear this bit. Writing a '0' has no effect. This bit is also cleared when **ResetStatusChange** is set. 0 = Resume is not completed 1 = Resume completed
17	PESC	PortEnableStatus Change	This bit is set when hardware events cause the **PortEnableStatus** bit to be cleared. Changes from HCD writes do not set this bit. The HCD writes a '1' to clear this bit. Writing a '0' has no effect. 0 = No change in **PortEnableStatus** 1 = Change in **PortEnableStatus**

Figure 7.2.50 HcRhPortStatus[1:2] Register (1/4)

Bits	Mnemonic	Field Name	Description
16	CSC	ConnectStatus Change	This bit is set whenever a connect or disconnect event occurs. The HCD writes a '1' to clear this bit. Writing a '0' has no effect. If **CurrentConnectStatus** is cleared when a **SetPortReset, SetPortEnable**, or **SetPortSuspend** write occurs, this bit is set to force the driver to re-evaluate the connection status since these writes should not occur if the port is disconnected. 0 = No change in **CurrentConnectStatus** 1 = Change in **CurrentConnectStatus** Note: If the **DeviceRemovable[NDP]** bit is set, this bit is set only after a Root Hub reset to inform the system that the device is attached.
15:10		Reserved	
9	LSDA	LowSpeedDevice Attached	(read) **LowSpeedDeviceAttached** This bit indicates the speed of the device attached to this port. When set, a Low Speed device is attached to this port. When clear, a Full Speed device is attached to this port. This field is valid only when the **CurrentConnectStatus** is set. 0 = Full speed device attached 1 = Low speed device attached (write) **ClearPortPower** The HCD clears the **PortPowerStatus** bit by writing a '1' to this bit. Writing a '0' has no effect.
8	PPS	PortPowerStatus	(read) **PortPowerStatus** This bit reflects the port's power status, regardless of the type of power switching implemented. This bit is cleared if an overcurrent condition is detected. HCD sets this bit by writing **SetPortPower** or **SetGlobalPower**. HCD clears this bit by writing **ClearPortPower** or **ClearGlobalPower**. Which power control switches are enabled is determined by **PowerSwitchingMode** and **PortPortControlMask[NDP]**. In global switching mode (**PowerSwitchingMode**=0), only **Set/ClearGlobalPower** controls this bit. In per-port power switching (**PowerSwitchingMode**=1), if the **PortPowerControlMask[NDP]** bit for the port is set, only **Set/ClearPortPower** commands are enabled. If the mask is not set, only **Set/ClearGlobalPower** commands are enabled. When port power is disabled, **CurrentConnectStatus**, **PortEnableStatus**, **PortSuspendStatus**, and **PortResetStatus** should be reset. 0 = Port power is off 1 = Port power is on (write) **SetPortPower** The HCD writes a '1' to set the **PortPowerStatus** bit. Writing a '0' has no effect. Note: This bit is always reads '1b' if power switching is not supported.
7:5		Reserved	

Figure 7.2.50 HcRhPortStatus[1:2] Register (2/4)

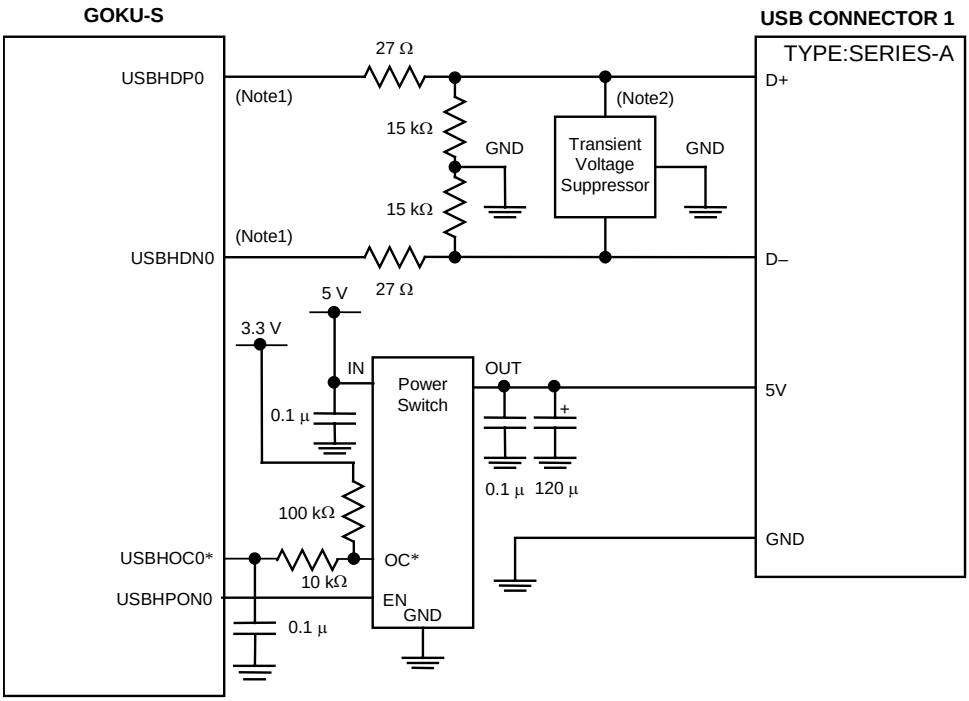
Bits	Mnemonic	Field Name	Description
4	PRS	PortResetStatus	(read) **PortResetStatus** When this bit is set by a write to **SetPortReset**, port reset signaling is asserted. When reset is completed, this bit is cleared when **PortResetStatusChange** is set. This bit cannot be set if **CurrentConnectStatus** is cleared. 0 = Port reset signal is not active 1 = Port reset signal is active (write) **SetPortReset** The HCD sets the port reset signaling by writing a '1' to this bit. Writing a '0' has no effect. If **CurrentConnectStatus** is cleared, this write does not set **PortResetStatus**, but instead sets **ConnectStatusChange**. This informs the driver that it attempted to reset a disconnected port.
3	POCI	PortOverCurrent Indicator	(read) **PortOverCurrentIndicator** This bit is only valid when the Root Hub is configured in such a way that overcurrent conditions are reported on a per-port basis. If per-port overcurrent reporting is not supported, this bit is set to 0. If cleared, all power operations are normal for this port. If set, an overcurrent condition exists on this port. This bit always reflects the overcurrent input signal 0 = No overcurrent condition. 1 = Overcurrent condition detected. (write) **ClearSuspendStatus** The HCD writes a '1' to initiate a resume. Writing a '0' has no effect. A resume is initiated only if **PortSuspendStatus** is set.
2	PSS	PortSuspend Status	(read) **PortSuspendStatus** This bit indicates the port is suspended or in the resume sequence. It is set by a **SetSuspendState** write and cleared when **PortSuspendStatusChange** is set at the end of the resume interval. This bit cannot be set if **CurrentConnectStatus** is cleared. This bit is also cleared when **PortResetStatusChange** is set at the end of the port reset or when the HC is placed in the USBRESUME state. If an upstream resume is in progress, it should propagate to the HC. 0 = Port is not suspended 1 = Port is suspended (write) **SetPortSuspend** The HCD sets the **PortSuspendStatus** bit by writing a '1' to this bit. Writing a '0' has no effect. If **CurrentConnectStatus** is cleared, this write does not set **PortSuspendStatus**; instead it sets **ConnectStatusChange**. This informs the driver that it attempted to suspend a disconnected port.

Figure 7.2.50 HcRhPortStatus[1:2] Register (3/4)

Bits	Mnemonic	Field Name	Description
1	PES	PortEnableStatus	(read) **PortEnableStatus** This bit indicates whether the port is enabled or disabled. The Root Hub may clear this bit when an overcurrent condition, disconnect event, switched-off power, or operational bus error such as babble is detected. This change also causes **PortEnabledStatusChange** to be set. HCD sets this bit by writing **SetPortEnable** and clears it by writing **ClearPortEnable**. This bit cannot be set when **CurrentConnectStatus** is cleared. This bit is also set, if not already, at the completion of a port reset when **ResetStatusChange** is set or port suspend when **SuspendStatusChange** is set. 0 = Port is disabled 1 = Port is enabled (write) **SetPortEnable** The HCD sets **PortEnableStatus** by writing a '1'. Writing a '0' has no effect. If **CurrentConnectStatus** is cleared, this write does not set **PortEnableStatus**, but instead sets **ConnectStatusChange**. This informs the driver that it attempted to enable a disconnected port.
0	CCS	CurrentConnect Status	(read) **CurrentConnectStatus** This bit reflects the current state of the downstream port. 0 = No device connected 1 = Device connected (write) **ClearPortEnable** The HCD writes a '1' to this bit to clear the **PortEnableStatus** bit. Writing a '0' has no effect. The **CurrentConnectStatus** is not affected by any write. Note: This bit is always read '1' when the attached device is nonremovable (**DeviceRemoveable[NDP]**).

Figure 7.2.50 HcRhPortStatus[1:2] Register (4/4)

7.3 USB HOST Connection Example



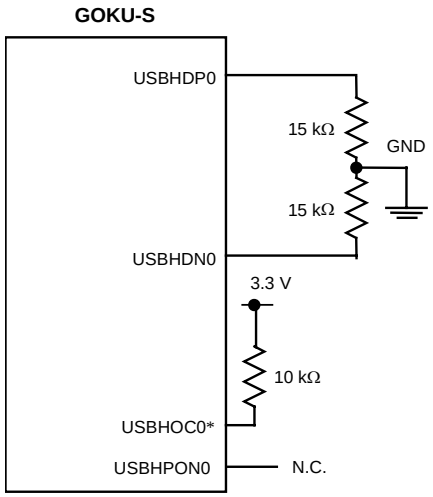
* Active-low signal

Example of power switch device: Texas Instruments TPS2052
MICREL MIC2526-1BM
MICREL MIC2536-1BN
Example of transient voltage suppressor device: Texas Instruments SN75240
Resistance precision: 5%
Resistance rating: Toshiba recommends ½ W for 27 Ω.
Capacitor: Toshiba recommends low ESR type (e.g. OS-CON) for the 120 μF capacitor.

Note 1: Don't apply the voltage over the absolute maximum rating.
Note 2: Not required by USB specifications.

The above figure is a USB port 1. Use a similar circuit for port 2.

When not using port 1, handle the unused pins as shown below. It applies to port 2.



8. USB Device Controller (Function 2)

8.1 Function Description

The USB Device Controller has an internal DMA Controller and supports Bus Master transfer and Slave transfer.

8.1.1 Feature

The USB Device Controller has the following features.

- Fully compliant with Universal Serial Bus Specification Rev 1.1.
- Has on-chip DMA that realizes Master transfer.
- On-chip DMA has a total of two channels: a Read channel and a Write channel. These channels can connect to the Endpoint 1 Register and Endpoint 2 Register, respectively.
- Bridge block contains 4-word FIFO.
- Supports 4-word Burst transfers.

8.1.2 Restrictions

- Only endpoints 1 and 2 can perform Master transfer.
- Please set an upper order value of a physical address from the start address to derive the end address to be used for Master transfers.
- Please do not perform Slave transfer to a UDC endpoint during Master transfer.
- Only Single Word transfers are possible in the case of Slave access.
- Only 32-bit transfers are possible in the case of Slave access.
- In the case of Master access, only Single Word transfers and 4-word Burst transfers are supported.
- Remote Wakeup is not supported.
- Isochronous transfer is not supported.

8.1.3 Device Requests

8.1.3.1 Standard requests

8.1.3.1.1 GET_STATUS request

This request automatically returns the specified reception status.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
10000000B 10000001B 10000010B	GET_STATUS	0	0 Interface Endpoint	2	Device, Interface or Endpoint Status

This request returns the following information to device requests in the Little Endian order.

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	Remote Wakeup	SelfPower
D15	D14	D13	D12	D11	D10	D9	D8
0	0	0	0	0	0	0	0

- RemoteWakeup — Returns the current Remote Wakeup settings.
  This device does not support RemoteWakeup.
  This bit is set and reset by the SET_FEATURE and CLEAR_FEATURE bits,respectively. The default value of this bit is 0.
- SelfPower — Returns the current power supply setting. Returns a value of either Self or BusPower according to the value set in the bmAttributes field inside the Config Descriptor.

This request returns two bytes of the value 00h to interface requests. This request returns the following information to endpoint requests in the Little Endian order.

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	HALT
D15	D14	D13	D12	D11	D10	D9	D8
0	0	0	0	0	0	0	0

- HALT — Returns HALT status of the selected endpoint.

8.1.3.1.2 CLEAR_FEATURE request

This request either clears or disables specific functions.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
00000000B 00000001B 00000010B	CLEAR_FEATURE	Feature Selector	0 Interface Endpoint	0	None

- Reception device

  FeatureSelector: 1 Disables the current RemoteWakeup setting.
  FeatureSelector: other than 1 Stalls

- Reception interface Stalls
- Reception endpoint

  FeatureSelector: 0 Clears the HALT of the applicable endpoint.
  FeatureSelector: other than 0 Stalls

Note: This request stalls when requests are made to non-existent endpoints.

8.1.3.1.3 SET_FEATURE request

This request either sets or enables specific functions.

bmRequestType	bRequest	wValue	WIndex	wLength	Data
00000000B 00000001B 00000010B	SET_FEATURE	Feature Selector	0 Interface Endpoint	0	None

- Reception device

  FeatureSelector: 1 Enables the current RemoveWakeup setting.
  FeatureSelector: other than 1 Stalls.

- Reception interface Stalls.
- Reception endpoint

  FeatureSelector: 0
  FeatureSelector: other than 0 Stalls.

Note: This request stalls when requests are made to non-existent endpoints.

8.1.3.1.4 SET_ADDRESS request

This request sets the device address. Subsequent requests respond by using the address set here. This request responds with the previous device address until the Status stage of this request ends normally.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
00000000B	SET_ADDRESS	Device Address	0	0	None

8.1.3.1.5 SET_DESCRIPTOR request

This request transmits the specified descriptor.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
10000000B	GET_ DESCRIPTOR	Descriptor Type and Descriptor Index	0 or Language ID	Descriptor Length	Descriptor

- Device — Transmits the device descriptor stored in the Descriptor RAM. The idProduct and bcdDevice fields are available as registers when using Descriptor ROM, so it is possible to rewrite only this area even if descriptor data is defined as ROM. When doing so, please access this register and define the data before connecting to the USB Host.
- Config — Transmits the Config Descriptor stored in Descriptor RAM. Continues to transmit the next Interface and Endpoint Descriptor that follows the Config Descriptor.
- String — Transmits the the String Descriptor of the index specified by the lower bytes of the wValue field.

Note: This request uses Get_Descriptor auto response to transmit whichever of wLength or Descriptor Length has the shortest descriptor.

8.1.3.1.6 SET_DESCRIPTOR request

This request either sets or enables specific functions.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
00000000B	SET_ DESCRIPTOR	Descriptor Type and Descriptor Index	0 or Language ID	Descriptor Length	Descriptor

Auto response for this request is not supported.

If, in response to an INT_SETUP interrupt, it is determined that the received request is the SET_DESCRIPTOR request, please confirm that the EP0_DSET_A bit of the DATASET Register is “1”, then extract the data. When complete, end the Status stage by accessing the EOP Register and writing “0” to the EP0_EOPB bit. The process is similar to that for Vendor requests, so refer to 8.1.3.3 Vendor requests (Class requests) for more information.

8.1.3.1.7 GET_CONFIGURATION request

This request transmits the configuration value of the current device.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
10000000B	GET_ CONFIG	0	0	1	Configuration Value

This request returns “0” when a device is not configured, or returns the configuration value of a device that is configured.

8.1.3.1.8 SET_CONFIGURATION request

This request sets the device configuration.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
00000000B	SET_CONFIG	Configuration Value	0	0	None

This request configures a device by using the value specified by the lower bytes of the wValue field.

8.1.3.1.9 GET_INTERFACE request

This request returns the AlternateSetting value set by the specified interface.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
10000001B	GET_INTERFACE	0	Interface	1	Alternate Setting

This request stalls if the specified interface does not exist.

8.1.3.1.10 SET_INTERFACE request

This request selects the AlternateSetting that resides in the specified interface.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
00000001B	SET_INTERFACE	AlternateSetting	Interface	0	None

This request stalls if the specified interface does not exist.

8.1.3.1.11 SYNCH_FRAME request

This request transmits the endpoint synch frame.

This device does not support this request.

8.1.3.2 Printer Class requests

The USB Device Controller does not support auto response Printer Class requests.

8.1.3.3 Vendor requests (Class requests)

The USB Device Controller does not support auto response for Vendor requests. If, in response to an INT_SETUP interrupt, the register containing a Device request is accessed and the received request is determined to be a Vendor request, then it is necessary to operate the USB Device Controller remotely and perform the corresponding process.

The following sections explain cases in which the Data phase is transmitted (Control Read) and received (Control Write).

8.1.3.3.1 Control Read request

bmRequestType	bRequest	wValue	wIndex	wLength	Data
110000xxB	Vendor Specific	Vendor Specific	Vendor Specific	Vendor Specific (other than 0)	Vendor Data

If the application receives an INT_SETUP interrupt, please judge the content of the received request using the bmRequestType, bRequest, wValue, wIndex, or wLength Register, then perform the process that corresponds to the request. The application must access the Setup_Received Register and signal acknowledgement of the INT_SETUP interrupt to the UDC after judging the request.

Once preparations for transmitting data are complete on the application side, please access the DATASET Register, confirm that the EP0_DSET_A bit is “0”, then write data to the endpoint 0. When transmitting data larger than the payload, it is necessary to poll the EP0_DSET bit of the DATASET Register and confirm it is “0” before writing data. (It is okay to use the INT_ENDPOINT0 interrupt signal for this purpose.) Please write “0” to the EP0 bit of the EOP Register after all data writing is complete. The UDC receives this data, and then automatically ends the Status stage.

Also, the UDC asserts the INT_STATUS interrupt when the Status stage ends normally. If an external application can comprehend that the Status stage ended normally, then please use this interrupt signal to manage the stage. A new Setup token may be received when the Status stage does not end normally or while the same stage is still in progress. When the INT_SETUP interrupt signal is asserted in this situation, setting the STAGE_ERROR bit of the EP0_STATUS Register to “1” signals the outside of the UDC that the Status stage did not end normally.

There are also cases in which a protocol during a USB Control Read transfer causes the Data phase to end at a data count that is less than the value indicated by wLength. If an application program has configured a process with only the wLength value, then it will not be able to function properly if the Host transitions to the Status stage before the expected data count are reached. In this situation, the INT_STATUSNAK interrupt signal can be used to signal the transition to the Status stage. (However, it is necessary to clear the mask of the STATUS_NAK bit using the Interrupt Control Register.) In the case of a Vendor request, such an event will not actually occur since the Reception buffer size will be set in the Host Controller by the driver. (Depending on the Host, please note that it may be necessary to use the software to control Standard Read requests since data sent from the device using an 8-byte payload will be recognized as a short packet and since the above situation may result until the payload size of the device is known.)

8.1.3.3.2 Control Write request

When there is no Data phase

bmRequestType	bRequest	wValue	wIndex	wLength	Data
010000xxB	Vendor Specific	Vendor Specific	Vendor Specific	0	None

If the application receives an INT_SETUP interrupt, please judge the content of the request using the bmRequestType, bRequest, wValue, wIndex, or wLength Register, then perform the process that corresponds to the request. The application must access the Setup_Received Register and signal acknowledgement of the INT_SETUP interrupt to the UDC after judging the request.

Please write "0" to the EP0 bit of the EOP Register if the process ended at the application side. The UDC receives this then automatically ends the Status stage.

When there is a Data phase

bmRequestType	bRequest	wValue	wIndex	wLength	Data
010000xxB	Vendor Specific	Vendor Specific	Vendor Specific	Vendor Specific (other than 0)	Vendor Data

If the application receives an INT_SETUP interrupt, please judge the content of the Device request using the bmRequestType, bRequest, wValue, wIndex, or wLength Register, then perform the process that corresponds to the request. The application must access the Setup_Received Register and signal acknowledgement of the INT_SETUP interrupt to the UDC after judging the request.

Once preparations for receiving data are complete on the application side, please access the DATASET Register, confirm that the EP0_DATASET is "1", and then read out data from the endpoint 0. When receiving data larger than the payload, it is necessary to poll the EP0_DSET bit of the DATASET Register and confirm it is "1" before reading out data from the next packet. (It is okay to use the INT_ENDPOINT0 interrupt signal for this purpose.) Please write "0" to the EP0 bit of the EOP Register after all data reading is complete. The UDC receives this data, then automatically ends the Status stage.

Also, the UDC asserts the INT_STATUS interrupt when the Status stage ends normally. If an external application can comprehend that the Status stage ended normally, then please use this interrupt signal to manage the stage. A new Setup token may be received when the Status stage does not end normally or while the same stage is still in progress. When the INT_SETUP interrupt signal is asserted in this situation, setting the STAGE_ERROR bit of the EP0_STATUS Register to "1" signals the outside of the UDC that the Status stage did not end normally.

The following figure shows the UDC control flow as seen from an application.

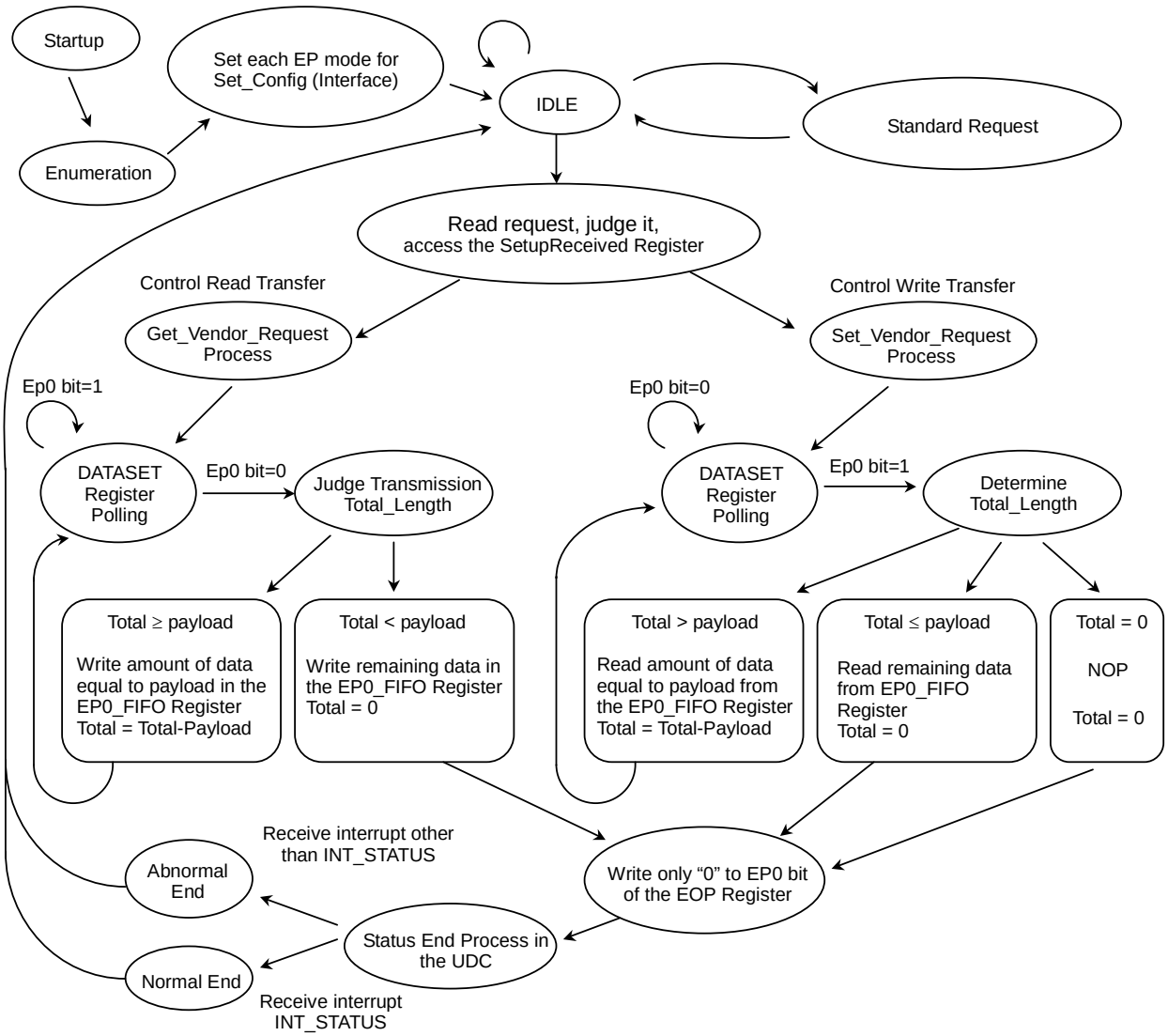


Figure 8.1.1 UDC Control Flow Seen From an Application

- The flow does not clarify special cases where overlapping Setup packets are received. Refer to the explanation in 8.1.4.4 Control transfer for more information pertaining to the flow.

- Requests are shown as a series flow, but it is possible to modularize the process into separate interrupts. Please contact us for any technical inquiries regarding the issue of sample programs.

8.1.4 Transfer Mode and Protocol Processes

The UDC uses hardware to automatically perform the following: receive packets; determine the address, endpoint, transfer mode; process errors; confirm the toggle bit/CRC of data reception packets; generate the toggle bit/CRC or data transmission packets; perform handshake responses.

8.1.4.1 Protocol overview

The USB specifies packet transaction formats. For both receive and transmit operations, the UDC processes USB data packets in hardware. The following paragraphs describe the fields that comprise a packet.

- SYNC field

  Every packet always starts with a SYNC field. The UDC uses it to synchronize incoming data with its local clock.

- Packet Identification field

  A PID field immediately follows the SYNC field in every USB packet. The UDC determines the PID type and judges the transfer type by decoding this code.

- Address field

  The UDC uses this field to confirm whether this function is specified from the Host. The UDC compares this field to the address set in the Address Register. If the addresses match, then the next process is performed. If they do not match, then the UDC ignores this token.

- Endpoint field

  This 4-bit field specifies a function if more than two sub-channels are required. The UDC can support up to seven endpoints, with the exception of control endpoints. The UDC ignores tokens for endpoints that are not supported.

- Frame number field

  This 11-bit field is incremented by the Host at each frame. This field follows the first SOF token that is sent at the beginning of each frame, and then a frame number is specified. When an SOF token is received, the UDC reads the contents of this field and sets the frame number in the Frame Register.

- Data field

  This field, which may be 0 – 1023 bytes in length becomes data that is in byte units. During reception, the UDC transfers only enough data for this data portion to the ENDPOINT, confirms CRC, asserts an interrupt signal, and then signals that data transfer to the ENDPOINT is complete. During transmission, the ENDPOINT data is transferred after the IN token, then the UDC adds a Data CRC field to the end of the data.

- CRC function

  A five-bit CRC is added to the token and a 16-bit CRC is added to the data. The UDC automatically compares the CRC of the received data to the CRC that was added, then automatically generates and transmits a CRC during transmission. Some transfer modes may not perform this CRC comparison, however.

8.1.4.2 Transfer mode

The UDC supports three transfer modes for the Full speed. Isochronous transfer is not supported.

- Full Speed devices

  Control transfer

  Interrupt transfer

  Bulk transfer

The following items describe the UDC operation for each transfer mode. In addition, the data flow up to the Endpoint is explained to each transfer operation. Please refer to Section 8.1.5 for more information regarding how to access Endpoint from PCI Bus.

8.1.4.2.1 Bulk transfer

Bulk transfer uses error detection and retries to ensure error-free transfers between the Host and a function. Three phases are basically used: Token, Data, Handshake. However, the Data phase is replaced by the Handshake phase due to flow control and Stall conditions. Only two phases are actually used as a result. The UDC retains the status of each endpoint and uses the hardware to perform flow control. It is possible to use the EPx_Status Register to confirm the status of each endpoint.

8.1.4.2.2 Bulk transmission mode

During transmission, bulk transfer is performed according to the following transaction format.

- Token: IN
- Data: Data0/Data1, NAK, STALL
- Handshake: ACK

Control flow

The control flow inside the UDC is as follows when an IN token is received.

(1) Receive a token packet, confirm the address, endpoint number or error. Then, check whether the transfer mode of the applicable endpoint is compatible to the IN token. Return to the Idle state if the transfer mode is not compatible.

(2) The UDC checks the ENDPOINT STATUS field in the EPx_STATUS register. If the endpoint status is "Invalid," the UDC returns to the Idle state. If the endpoint status is "Stall," the UDC returns a STALL handshake to the host and reverts to the Idle state. Next, the UDC accesses the DATASET register to determine whether the FIFO of the specified endpoint is ready to transmit a packet of data. If it isn't, the UDC returns a NAK handshake to the host. If it is ready, the UDC goes to the next step.

(3) The UDC moves the data from the FIFO to the Serial Interface Engine (SIE) and creates a data packet as per the USB protocol. When packetizing data, the UDC uses a register called Toggle Bit register, which is transparent to the user. In the course of this, the UDC obtains the byte count for the packet from the EPx_SIZE register and checks it against the maximum payload size for the endpoint. In case of more than maximum-sized data, the UDC forces a bit stuff error to abort the transfer, and then stall by writing "Stall" to the EPx_STATUS register.

(4) The UDC shifts data out of the FIFO and follows it with CRC bits.

(5) Completion of data transmission is indicated by the host. If the host receives a valid data packet, it responds with an ACK handshake. When the UDC sees an ACK handshake from the host, it proceeds with the following post-transmit operations:

- the Endpoint is cleared,
- the Data Set Register is cleared,
- the Toggle bit is updated and stands by for the next transfer,
- the Status is set to Ready,
- the UDC ends normally.

The ENDPOINT is now ready to accept the next data. If a timeout occurs before an ACK is received from the Host,

- the status is set to TX_ERR,
- the Endpoint address pointer is returned to its previous value.

Then, the Endpoint data is saved and the UDC waits for the next retry.

This flow is shown in Figure 8.1.2 on the following page.

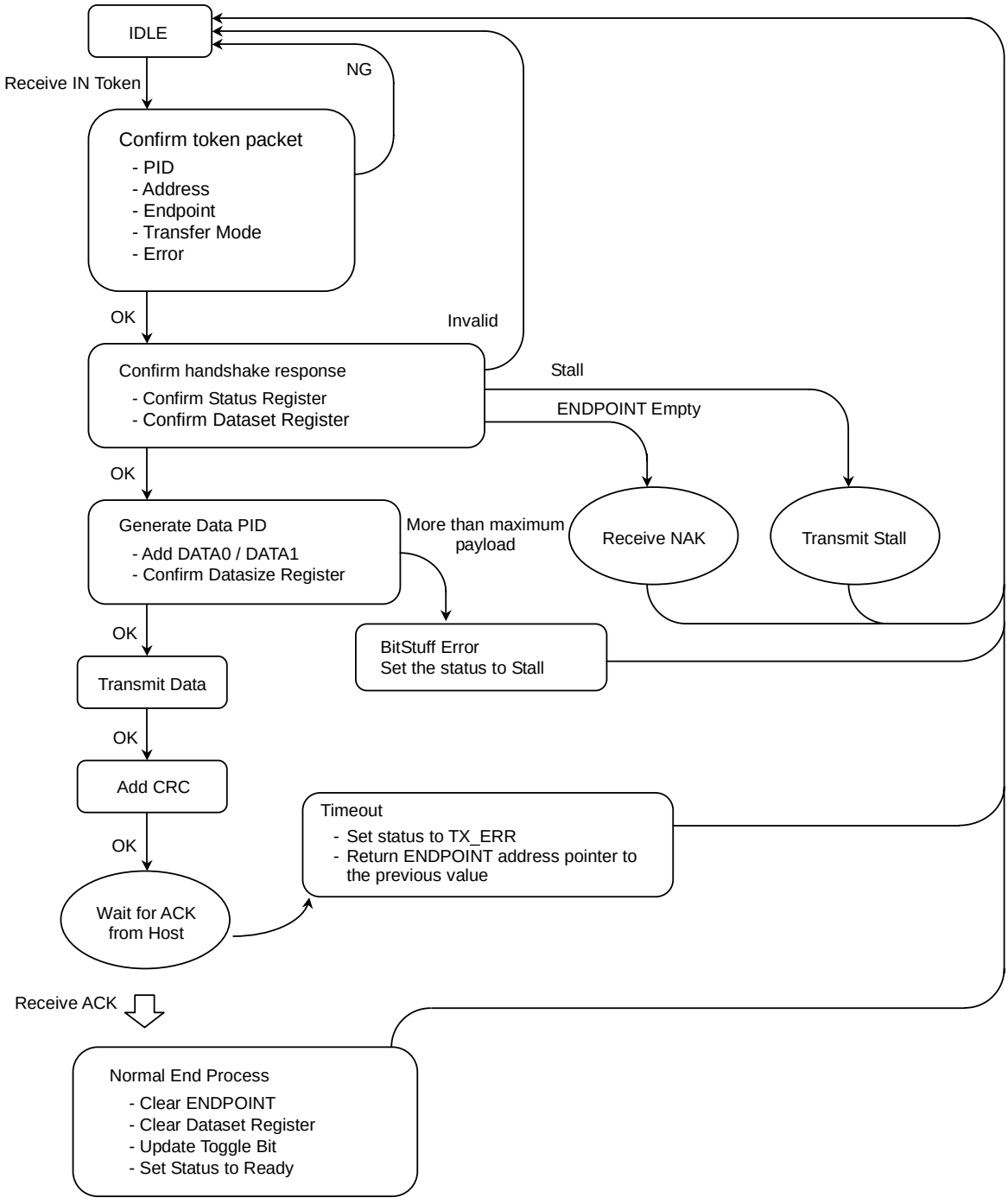


Figure 8.1.2 UDC Internal Control Flow
(Bulk Transfer (Transmission)/Interrupt Transfer (Transmission))

8.1.4.2.3 Bulk reception mode

During reception, bulk transfer is performed according to the following transaction format.

- Token: OUT
- Data: Data0/Data1
- Handshake: ACK, NAK, STALL

Control flow

The control flow inside the UDC is as follows when an OUT token is received.

(1) Receive a token packet, confirm the address, endpoint number or error. Then, check whether the transfer mode of the applicable endpoint is compatible to the OUT token. Return to the Idle state if the transfer mode is not compatible.

(2) Confirm the status of the Status Register.

Invalid state: Return to the Idle state.

Stall state: Return a Stall handshake as soon as the Data phase ends, return to the Idle state, and then destroy the data.

Confirm the Endpoint status. If preparations were not made for storing one packet of data, destroy the data just transferred, return an NAK handshake right after the Data phase and return to the Idle state.

(3) Receive a data packet. Transfer data from the SIE inside the UDC to the Endpoint. Confirm the data count transferred. If the size of the data exceeds the maximum payload size for each endpoint, the Status is set to Stall then returns to the Idle state. No ACK handshake is returned at this time.

(4) After the data is completely transferred to the Endpoint, the calculated CRC and the transferred CRC are compared. If they do not match, then RX_ERR is set in Status, and then returns to the Idle state without returning ACK. The USB Host attempts a retry, then Status changes to Data In if the next data is received normally. Also, if the data toggle does not match, then it will be judged that the Host was not able to receive ACK in the previous transfer. The current transfer will be interpreted as a retry of the transfer prior to the current one, data will be destroyed, Status will become RX_ERR, ACK will be returned to the Host, and will then return to the Idle state. The next data can be received since the Endpoint address pointer returns to the previous value.

(5) If CRC and the toggle match and transfer end normally, return the ACK handshake and perform the following process inside the UDC.

- Set the transfer data count in the Data Size Register.
- Set the Data Set Register.
- Update the Toggle bit and prepare for the next data.
- Set the Status to Data In.

The UDC now ends normally.

This flow is shown in Figure 8.1.3 on the following page.

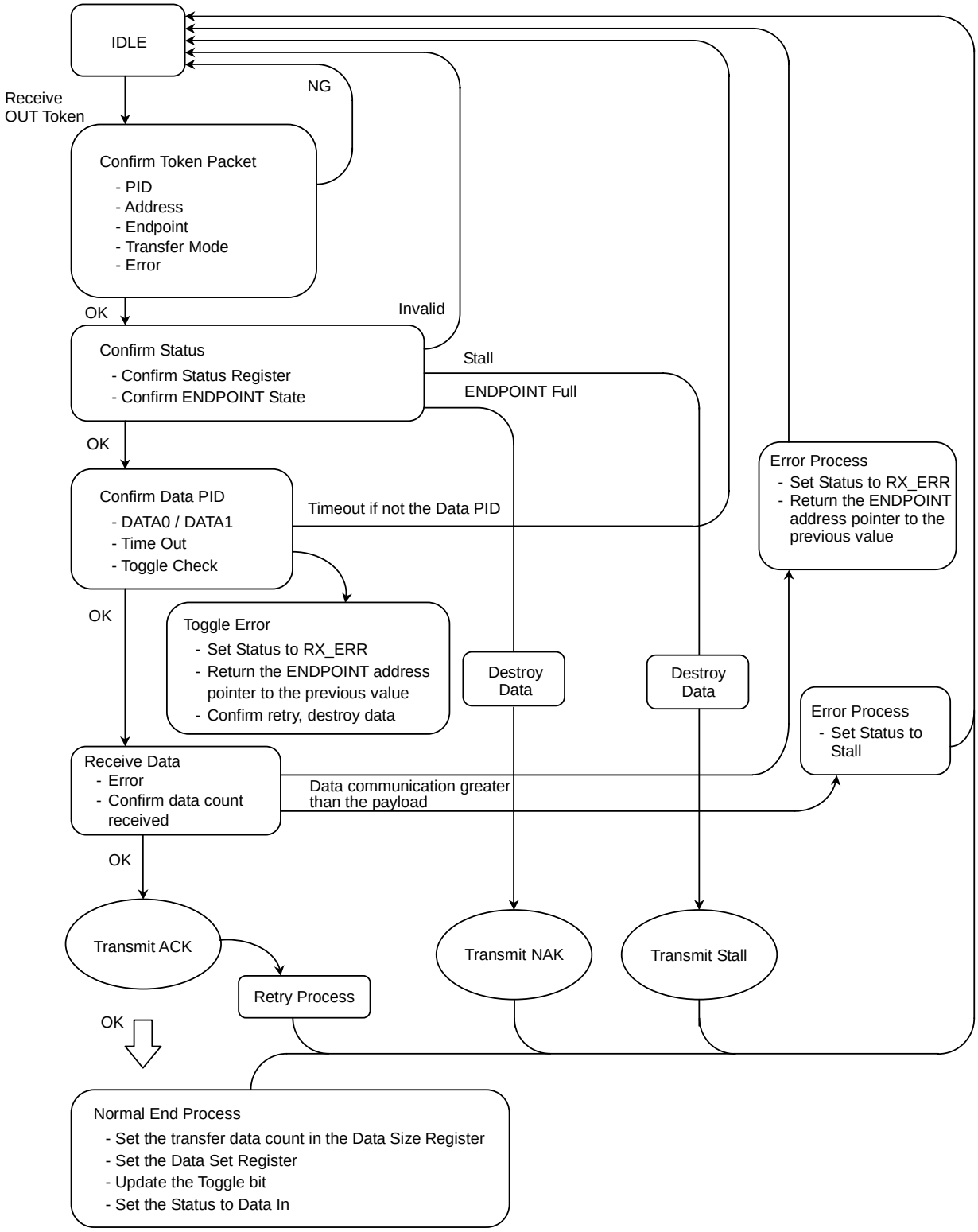


Figure 8.1.3 UDC Internal Control Flow (Bulk Transfer (Reception))

8.1.4.3 Interrupt transfer

Interrupt transfers use the same transaction format as Transmission Bulk transfers. The UDC hardware settings and responses when transferring using the Toggle bit are identical to those for Transmission Bulk transfer. In the case of Interrupt transfers, it is possible to perform transfers without using the Toggle bit. In this situation, the Toggle bit is updated and the transfer ends normally even if no ACK handshake is received from the Host. The UDC clears the ENDPOINT for the next transfer.

8.1.4.3.1 Interrupt Transmission mode (Toggle mode)

UDC operation is identical to that for the Bulk Transmission mode. Please refer to 8.1.4.2.3.

8.1.4.3.2 Interrupt Transmission mode (NOT Toggle mode)

This mode is basically identical to the Bulk Transmission mode, but the process differs in this case when the ACK handshake is not received from the Host.

After sending a data packet, the following occurs when the ACK handshake is received from the Host:

- the ENDPOINT is cleared,
- the Data Set Register is cleared,
- the Toggle bit is updated and stands by for the next transfer,
- the Status is set to Ready,
- the UDC ends normally.

The ENDPOINT is ready to receive the next data.

If a timeout occurs before ACK is received from the Host,

- the ENDPOINT is cleared,
- the Data Set Register is cleared,
- the Toggle bit is updated and is ready for the next data,
- Status is set to TX_ERR.

The only difference here is the change of Status setting.

8.1.4.4 Control transfer

Control transfer consists of the three following stages.

- Setup stage
- Data stage
- Status stage

The Data stage may be omitted. Each stage consists of either single or multiple transactions. The UDC performs each transaction process while using the hardware to manage the three stages. There are three types of Control transfer depending on the absence/presence of the Data stage and the transfer direction:

- Control Read transfer
- Control Write transfer
- Control Write transfer (no Data stage)

The three above transfer sequences are illustrated in Figure 8.1.5, Figure 8.1.6 and Figure 8.1.7, respectively.

Furthermore, the UDC uses the hardware to automatically respond to Standard requests. The CPU must intervene to control the UDC for Class requests and Vendor requests other than optional Standard requests.

Each item below describes the UDC internal control flow and the control flow when the CPU intervenes.

8.1.4.4.1 Setup stage

Except for the Token ID becoming “Setup,” the Setup stage is identical to a Transmission Bulk transaction. However, the UDC control flow differs.

- Token: Setup
- Data: Data0
- Handshake: ACK

Control flow

Following is the UDC internal control flow for when a Setup token is received.

(1) Receive a Setup token. Confirm the address, endpoint number, or error. Check whether the applicable endpoint is in the Control transfer mode. Return to the Idle state if it is not.

(2) Confirm the state of the Status Register. This register only returns to the Idle state when it is in the Invalid state. In the case of Bulk transfer, data reception was possible depending on the value of the Status Register or the Endpoint state. In the case of the Setup stage however, Status is returned to Ready, access to the Endpoint from the CPU is restricted, the endpoint 0 is cleared, and preparations are made for the next Data phase in all situations. When the CPU accesses the Setup Received Register inside the UDC, receipt of a Device request is acknowledged and the endpoint0 accesses restrictions from the CPU are cancelled. This is done to make it possible to receive a new request before the

previous Device request process ends normally.

(3) Receive a Data packet. Transfer an 8-byte Device request from the SIE inside the UDC and transfer data to the following Request Registers.

- bmRequestType Register
- bRequest Register
- wValue Register
- wIndex Register
- wLength Register

(4) Compare the calculated CRC to the transferred CRC after the last data is transferred to the Endpoint. If the CRC values do not match, set RX_ERR in Status, and then return to the Idle state without returning an ACK handshake to the Host. The Host will perform a retry.

(5) If CRC and the Toggle match and operation end normally, return an ACK handshake to the Host.

- Determine the control authority of the received Device request. If a request requires control by the software, then assert an INT_SETUP interrupt and signal outside the UDC that a request was received. The INT_SETUP interrupt is not asserted for auto responses performed by the hardware.
- Make preparations for the next stage according to the Stage control flow.
- Set the Status to Data In.
- Set the Toggle bit to “1”.

This flow is shown in Figure 8.1.4 UDC Internal Control Flow Diagram (Setup Stage).

8-byte data that is transferred by this Setup stage becomes a Device request. The CPU must perform a process that corresponds to a device request. The UDC detects only the following content from the 8-byte data, and then uses the hardware to perform stage management.

- Absence/Presence of the Data stage
- Direction of the Data stage

This flow is used to judge Control Read transfer, Control Write transfer and Control Write (Without Data Phase).

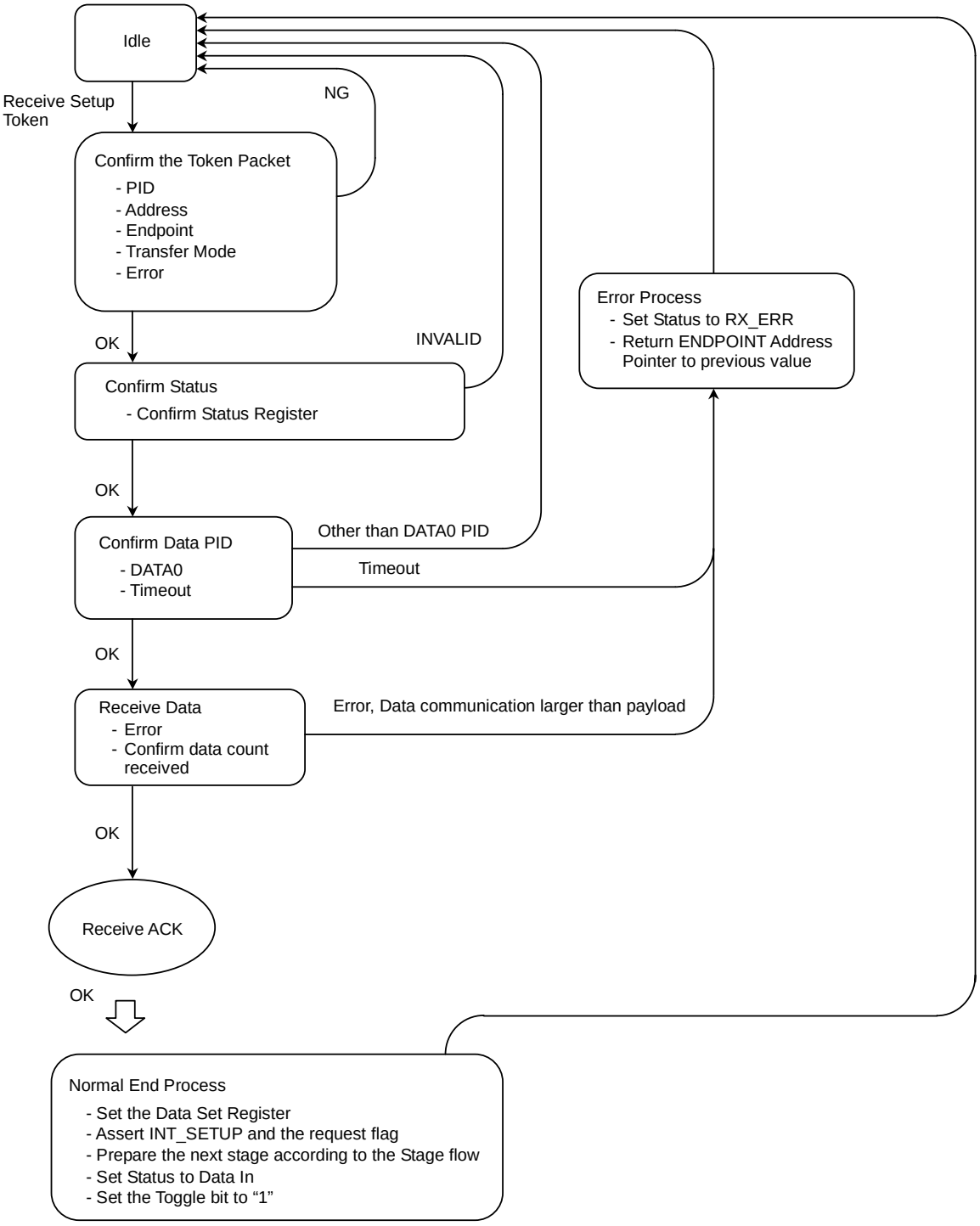


Figure 8.1.4 UDC Internal Control Flow Diagram (Setup Stage)

8.1.4.4.2 Data stage

The Data stage consists of either a single transaction or multiple transactions based on the Toggle sequence. The transactions have a format that is identical to that of Transmission or Reception Bulk transactions, but differ in the following areas.

- The Toggle bit starts from "1" when the Setup stage starts.
- An In or Out token is compared to the direction bit in a Device request and is judged to be correct or incorrect. A received token with the reverse transfer direction is recognized as the Status stage.
- The INT_ENDPOINT0 interrupt is asserted.

8.1.4.4.3 Status stage

The Status stage consists of an IN or OUT token followed by a packet with a data length of 0 along with Data1 PID and a handshake. A transaction with a different direction than the previous stage is used. Following are some example combinations.

- Control Read transfer: OUT
- Control Write transfer: IN
- Control Write transfer (without Data phase): IN

The UDC processes the Status stage based on the flow control of the internal Control transfer. It is necessary for the CPU to write "0" to the EP0 bit of the EOP Register at the end of the process so the Status stage can end normally. The details of the Status stage are described below.

8.1.4.4.4 IN Status stage

Transactions for the In Status stage have the following format.

- Token: IN
- Data: Data1 (Data length: 0), NAK, STALL
- Handshake: ACK

Control flow

Following is the process flow of the IN Status stage inside the UDC.

(1) Receive a token packet. Confirm the address, endpoint number or error. Return to the Idle state if the result is incorrect. Proceed to the next step if the Status stage is enabled based on the stage control flow inside the UDC.

(2) Confirm the state of the Status register.
- Invalid state: Return to the Idle state
- Stall state: Return a Stall handshake, return to the Idle state.

Confirm whether the EOP Register is accessed from outside the UDC. If it has not been accessed, then return an NAK handshake in order to continue Control transfer, then return to the Idle state.

(3) Transmit a data packet with a length of 0 and CRC when it has been confirmed that the EOP Register was accessed.

(4) When an ACK handshake is received from the Host,

- set Status to Ready,
- assert an INT_STATUS interrupt.

The process ends normally at this point. If a timeout occurs before an ACK handshake is received from the Host,

- Set TX_ERR in the Status Register and return to the Idle state. Wait for a retry of the Status stage.

If a new Setup stage is started before the Status stage ends normally, the UDC sets a Stage error in the Status Register.

8.1.4.4.5 OUT Status stage

Transactions for the OUT Status stage inside the UDC have the following format.

(1) Receive a token packet. Confirm the address, endpoint number, or error. Return to the Idle state if the result is incorrect. Proceed to the next step if the Status stage is enabled based on the stage control flow inside the UDC.

(2) Confirm the state of the Status register.

- Invalid state: Return to the Idle state.
- Stall state: Destroy the Data packet, return a Stall handshake, return to the Idle state.

Confirm whether the EOP Register was accessed from outside the UDC. If it was not accessed, return an NAK handshake to continue the Control transfer, and then return to the Idle state.

(3) When it has been acknowledged that the EOP Register was accessed, then a 0 data packet and CRC is received.

(4) Transmit an ACK handshake to the Host if there are no errors in the data.

- Set the status to Ready.
- Assert an INT_STATUS interrupt.

This stage ends normally.

If there was an error in the data, do not return an ACK handshake.

- Set RX_ERR in the Status Register, return to the Idle state, and then wait for a Status stage retries.

The UDC sets a Stage error in the Status Register if a new Setup stage starts before the Status stage ends normally.

8.1.4.4.6 Stage management

The UDC uses the hardware to manage the progress of each stage of Control transfers. Transitions between each stage are performed by receiving a token from the USB Host or by the CPU using the software to access a register. Consequently, each stage of the Control transfer must proceed while remaining linked to the software. In addition, the UDC detects only the following contents from 8-byte Setup stage data, determines the Control transfer type, and performs stage management.

- Presence/Absence of the Data stage
- Direction of the Data stage

These contents are used to judge Control Read transfer, Control Write transfer, and Control Write transfer (Without the Data stage).

The conditions for transitioning between stages during each type of Control transfer are indicated below on the next page.

An NAK handshake is returned and Busy is signaled to the USB Host if a token corresponding to the next stage is received from the Host before the state inside the UDC changes to the next state. In addition, the current process is suspended and the stage inside the UDC changes to the Setup stage when a Setup token is received from the Host, regardless of the Control transfer type or current state. Even if the CPU is currently executing the previous Control transfer, a new INT_SETUP interrupt must be responded to when received.

Stage Transition Conditions for Control Read Transfer

- Received Setup token from the Host
  - The Setup stage inside the UDC starts.
  - Normally receive the request data, determine its type, and assert the INT_SETUP interrupt to outside the UDC.
  - Change to the Data stage inside the UDC.
- Received IN token from the Host
  - The CPU retrieves request from the Request Register in response to an INT_SETUP interrupt.
  - Determine the request type, access the SetupReceived Register in order to signal to the UDC that an INT_SETUP interrupt was acknowledged.
  - Monitor the EP0 bit of the Data set Register and write data to the ENDPOINT according to the content of the Device request.
  - The UDC sets the EP0 bit of the Data Set Register either when data equivalent to the payload is set in the ENDPOINT or when the CPU invokes Short Packet transfer using the EOP Register.
  - The UDC transfers the data set in the Endpoint in response to the IN token.
  - The CPU writes “0” to the EP0 bit of the EOP Register when the process ends.
  - Change to the Status stage inside the UDC.
- Received OUT token from the Host
  - Return ACK to the OUT token, change to the Idle state inside the UDC.
  - Assert an INT_STATUS interrupt to the outside.

Figure 8.1.5 below shows these status transitions.

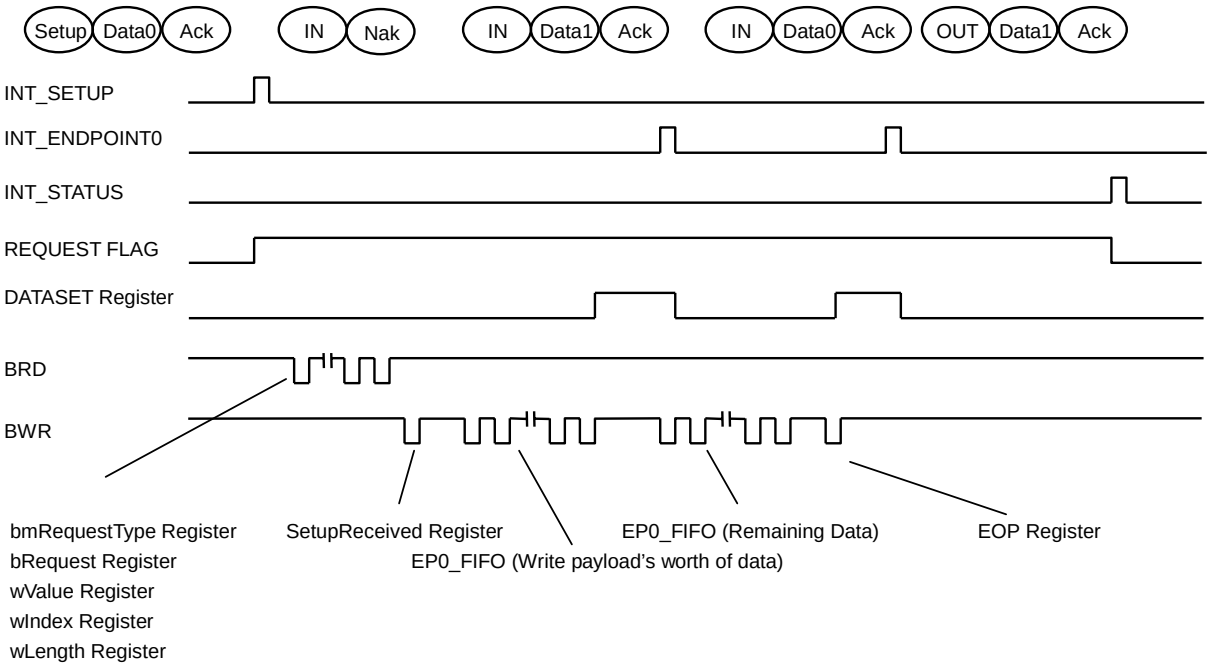


Figure 8.1.5 UDC Internal Control Flow Diagram (Control Read Transfer)

Stage Transition Conditions for Control Write Transfer

- Received Setup token from the Host
  - The Setup stage inside the UDC starts.
  - Normally receive the request data, determine its type, and assert the INT_SETUP interrupt to outside the UDC.
  - Change to the Data stage inside the UDC.
- Received OUT token from the Host
  - The CPU retrieves request from the Request Register in response to an INT_SETUP interrupt.
  - The CPU determines the request type, access the SetupReceived Register in order to signal to the UDC that an INT_SETUP interrupt was acknowledged.
  - Normally receive the Data phase data, and then set the EP0 bit of the Data Set Register.
  - The CPU retrieves data inside the ENDPOINT when the Data Set Register is set.
  - The CPU processes the received data along with the Device request.
  - The CPU writes “0” to the EP0 bit of the EOP Register when the process ends.
  - Change to the Status stage inside the UDC.
- Received IN token from the Host
  - Return a 0-data Data packet to the IN token, change to the Idle state inside the UDC.
  - Assert an INT_STATUS interrupt to the outside when ACK is received for a 0-data packet.

Figure 8.1.6 below shows these status transitions.

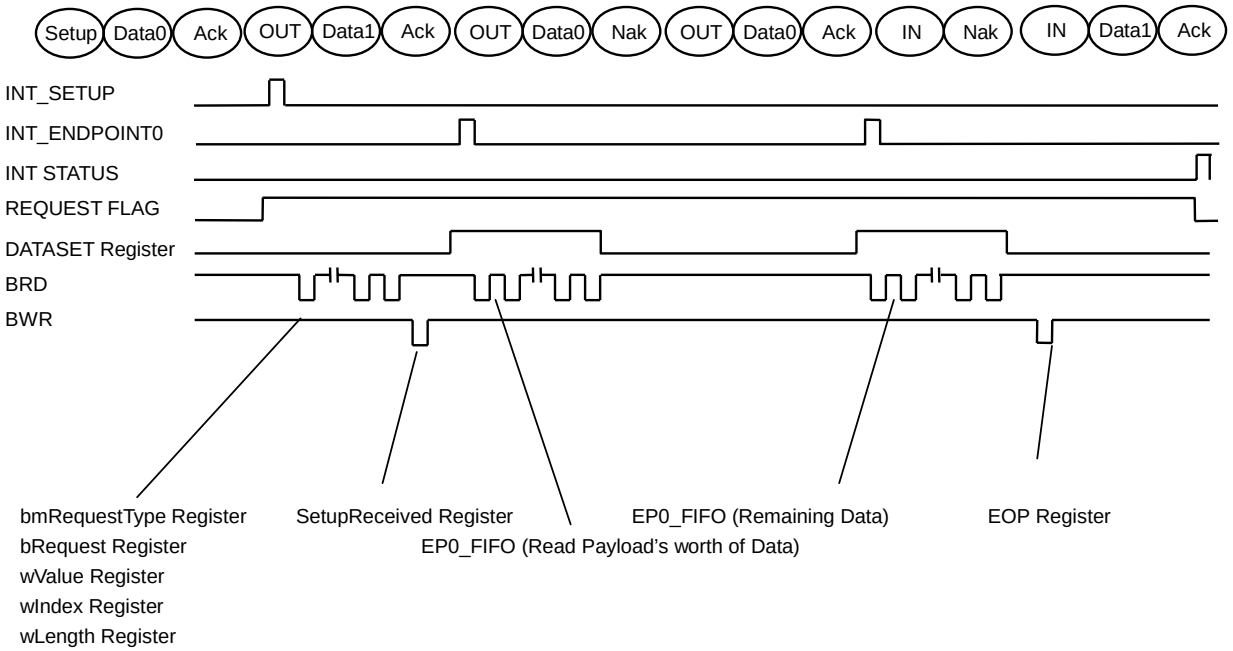


Figure 8.1.6 UDC Internal Control Flow Diagram (Control Write Transfer)

During Control Read transfer, the number of Data stage transactions does not necessarily match the data count specified by a Device request. It is therefore possible for the CPU to use an INT_STATUSNAK interrupt and proceed with a process. When using a Class or Vendor request however, it is not necessary to use this interrupt when using a Class or Vendor request if the wLength value and the Data transfer count in the Data phase are always made to match. When the data of the Data stage is unknown, it is possible to confirm the data count currently received by accessing the Data Size Register.

Stage Transition Conditions for Control Write Transfer (Without the Data Phase)

- Received Setup token from the Host
  - The Setup stage inside the UDC starts.
  - Normally receive the request data, determine its type, and assert the INT_SETUP interrupt to outside the UDC.
  - Change to the Data stage inside the UDC.
- Received In token from the Host
  - The CPU retrieves request from the Request Register in response to an INT_SETUP interrupt.
  - Determine the request type, access the SetupReceived Register in order to signal to the UDC that an INT_SETUP interrupt was acknowledged.
  - The CPU processes the received data along with the Device request.
  - The CPU writes "0" to the EP0 bit of the EOP Register when the process ends.
  - Change to the Status stage inside the UDC.
  - Return a 0-data Data packet to the IN token, change to the Idle state inside the UDC.
  - Assert an INT_STATUS interrupt to the outside when ACK is received for a 0-data packet.

Figure 8.1.7 below shows these status transitions.

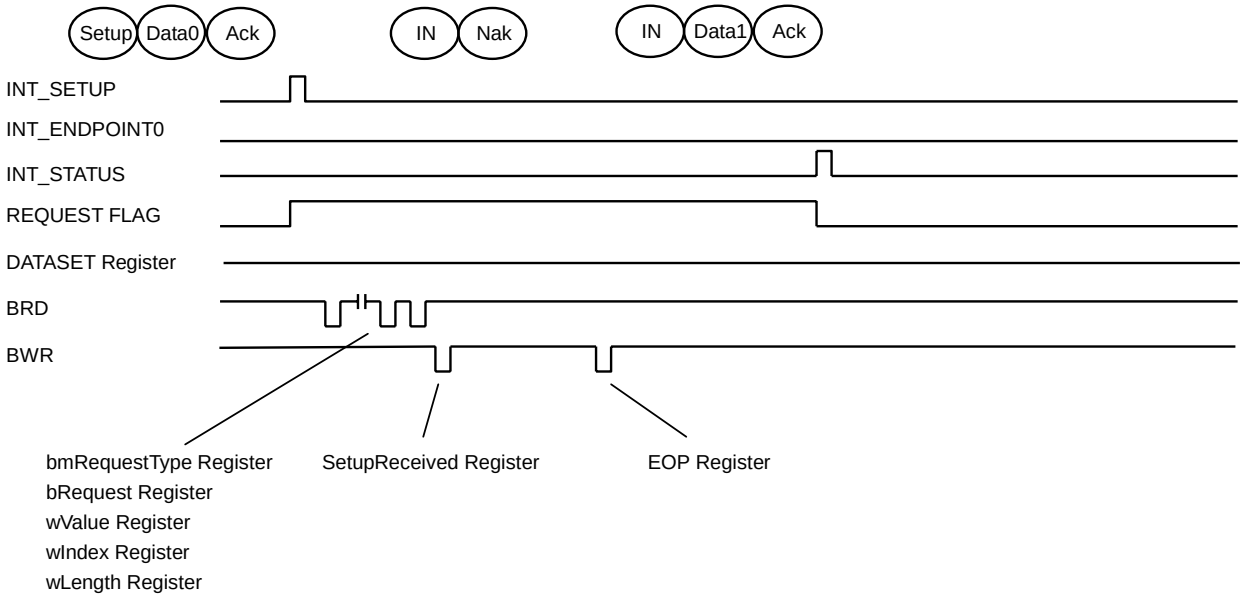


Figure 8.1.7 UDC Internal Control Flow (Control Write Transfer without Data Phase)

8.1.5 Accessing Bus Interface and Endpoint

This UDC provides the following busses as interfaces.

- CPU Bus I/F (8-bit Asynchronous Bus) Endpoint 0, 3
- Master Bus (DMA Bus) I/F (8-bit Asynchronous Bus) Endpoints 1, 2

Endpoint 0 is for exclusive use by the CPU Bus since the CPU processes Device request responses.

8.1.5.1 CPU Bus Interface

The UDC has two types of Endpoint access: Single Packet and Dual Packet. The Single Packet mode uses Endpoint capacity implemented by the hardware as one large Endpoint. The Dual Packet mode divides Endpoint capacity into two sections and uses them as independent Endpoints. The bus can be used efficiently since transfer with the Endpoint can be performed even while the UDC is transmitting/receiving data to/from the USB Host. However, Control transfer is only supported by the Single Packet mode.

It is necessary to fix the EPx_SINGLE signal of the endpoint used by the Dual Packet mode to "0". The Endpoint Register operates in the Single mode when this signal is fixed to "1".

Example: When endpoint 1 is used by a payload 64-byte Dual Packet.

EP1_FIFO Size: 128 bytes are available
EP1_SINGLE Signal: Fixed to "0"
Set EP1 Descriptor
Direction: Any direction
Max. payload size: 64 bytes
Transfer mode: Any mode

8.1.5.1.1 Single Packet mode

This section describes the data sequence of the Single Packet mode when using CPU Bus I/F. Figure 8.1.8 shows the reception sequence and Figure 8.1.9 shows the transmission sequence. The explanation in this item is centered on Endpoint access. Please refer to Section 8.1.4 for information regarding USB Host and the data sequence.

For endpoint 0, the mode cannot be changed since the endpoint is only used in the Single Packet Mode. Setting the EPx_SINGLE Register makes it possible to switch between the Single Packet mode for endpoints 1 – 3 and Dual Packet mode for endpoints 1 – 2. Please do not change the mode when a transfer is in progress.

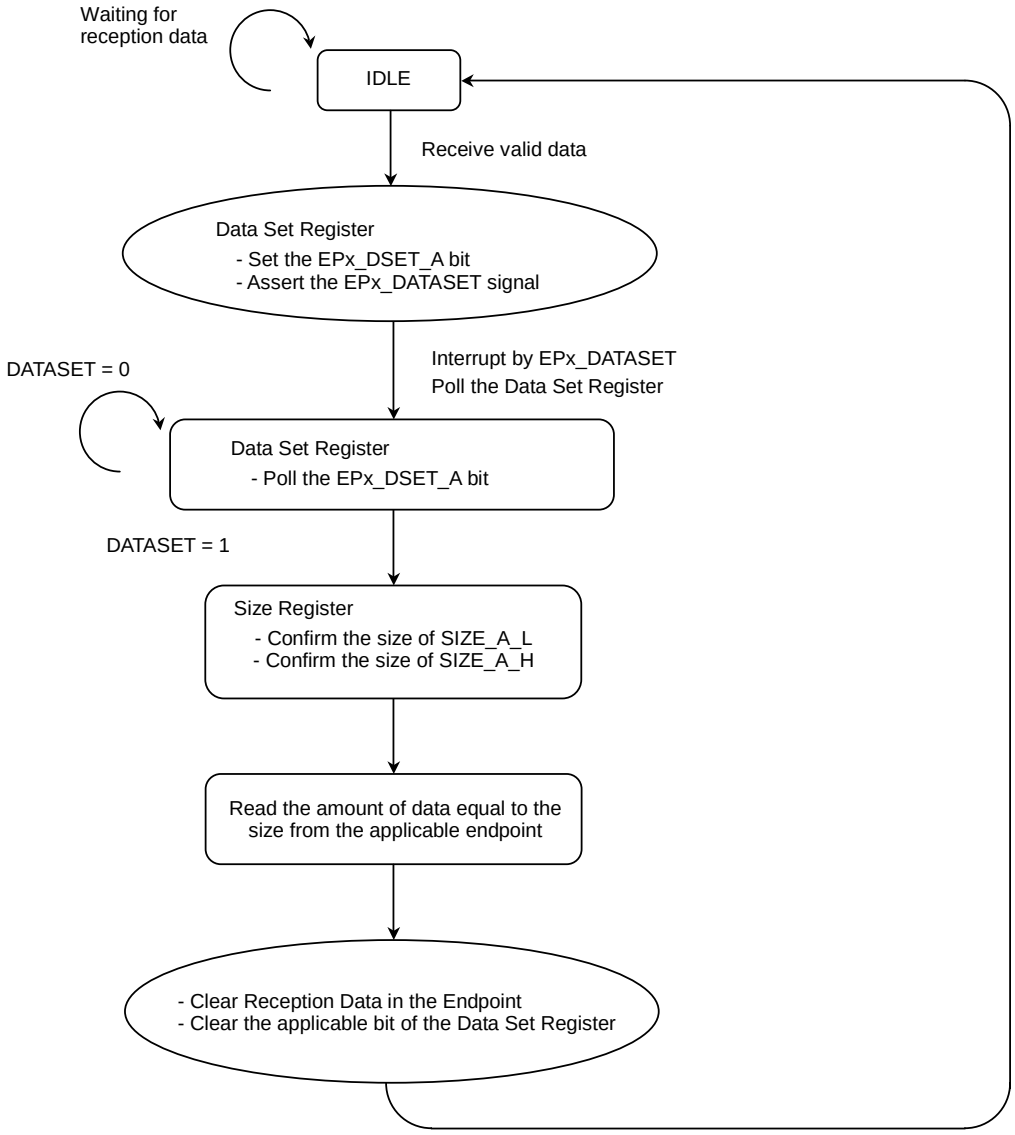


Figure 8.1.8 Reception Sequence in the Single Packet Mode

The following figure shows the transmission sequence when in the Single Packet mode.

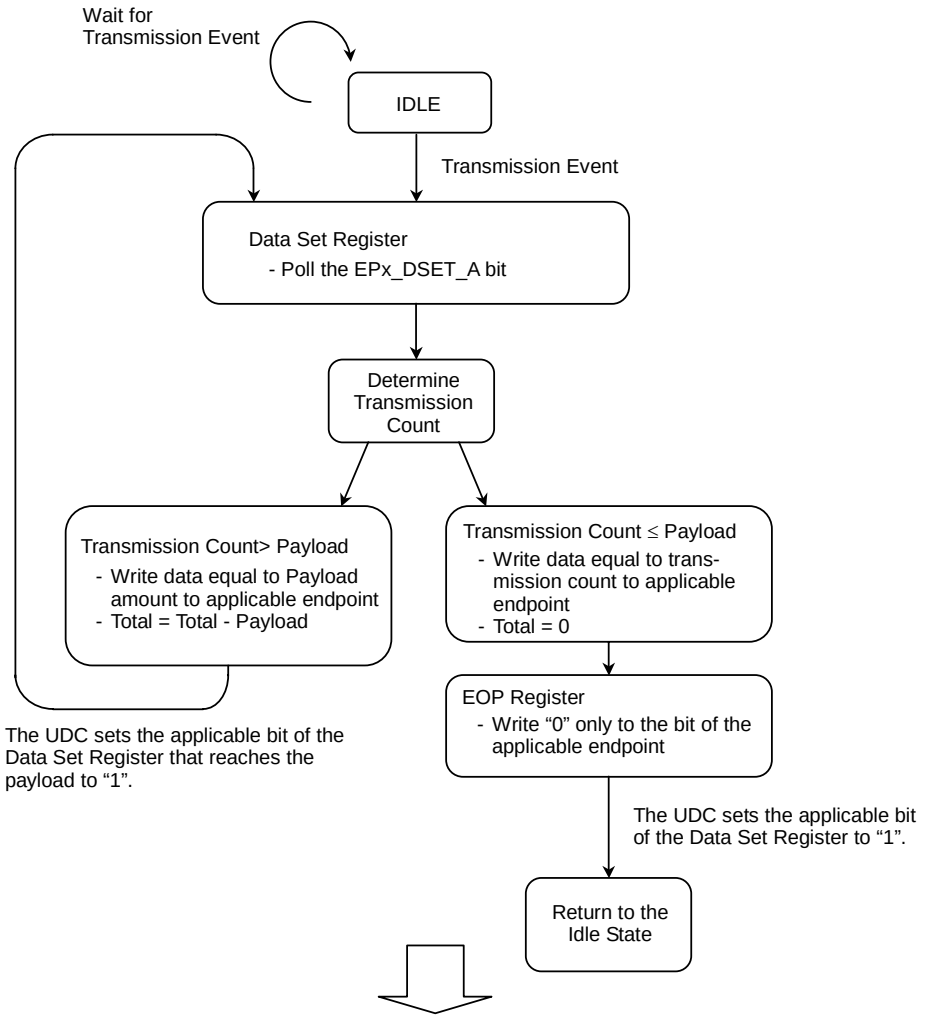


Figure 8.1.9 Transmission Sequence in the Single Packet Mode

8.1.5.1.2 Dual Packet mode

The Dual Packet mode divides the Endpoint into two independent packets, A and B, and uses the hardware to control the packets in order. This mode can transmit/receive data with the USB Host and transfer data on the PCI Bus simultaneously.

When reading out data from the Receive Endpoint, it is necessary to confirm the state of the two packets while taking the priority into consideration. Even if received data is held in two packets, the Endpoint that can be accessed has two packets. Therefore, the UDC outputs data in order from the previously received data. A Data Size Register is available for the A packet and the B packet, so it is necessary for the CPU to use the PACKET_ACTIVE bit, confirm which packet was accessed first, then recognize the data count of the packet that was received first. The packet whose PACKET_ACTIVE bit is set to “1” is the packet that was received first. The A packet and B packet always set the data alternately.

The following figure shows this sequence.

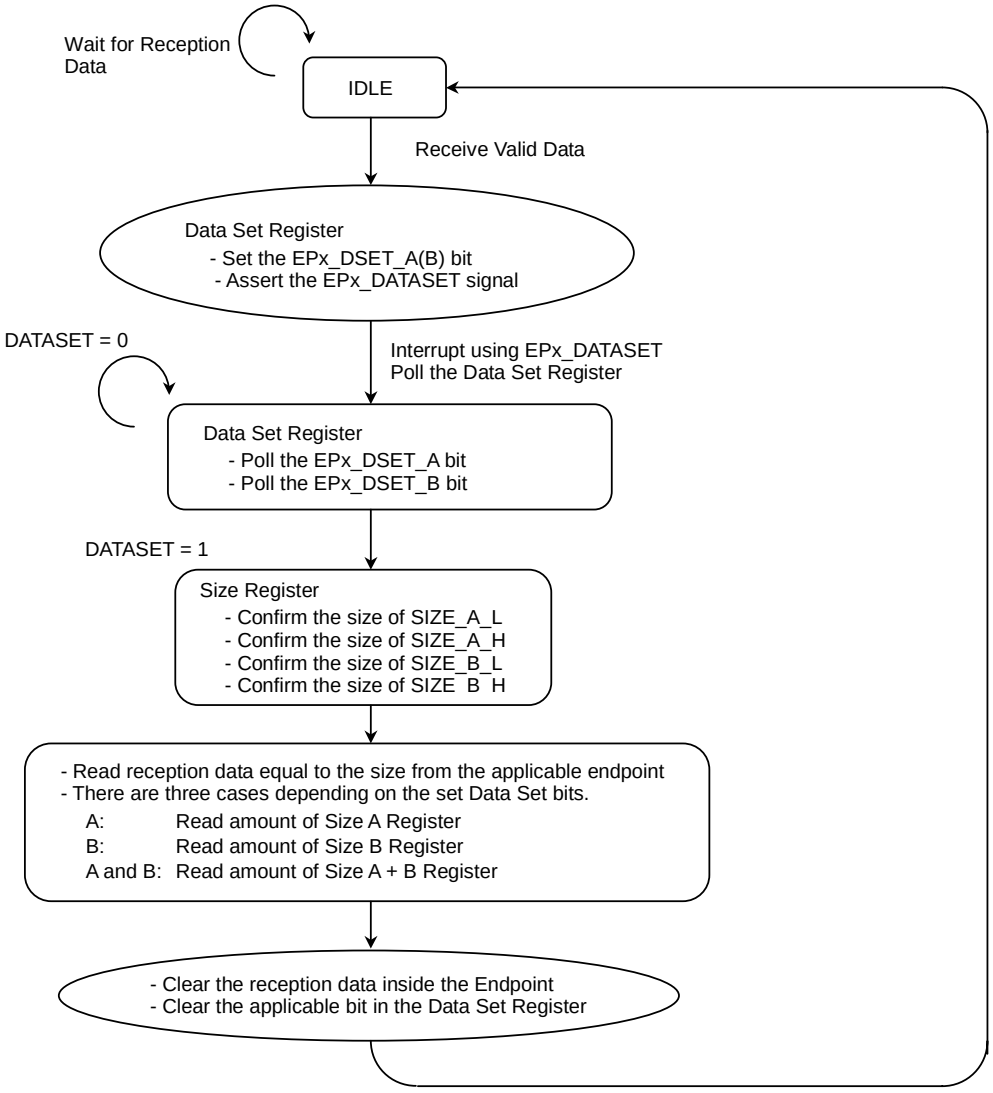


Figure 8.1.10 Reception Sequence in the Dual Packet Mode

When writing data to the Endpoint in a manner similar to that during transmission, it is necessary to confirm the state of the two packets and take the transfer order of Packets A and B into account before performing the data transfer. Look at the PACKET_ACTIVE bit to determine whether to set the transfer data count in Packet A or B. The packet whose PACKET_ACTIVE bit is “0” is the packet that will be transferred now.

Please carefully note that the logic of the PACKET_ACTIVE bit during reception and during transmission is reversed.

The following figure shows this sequence.

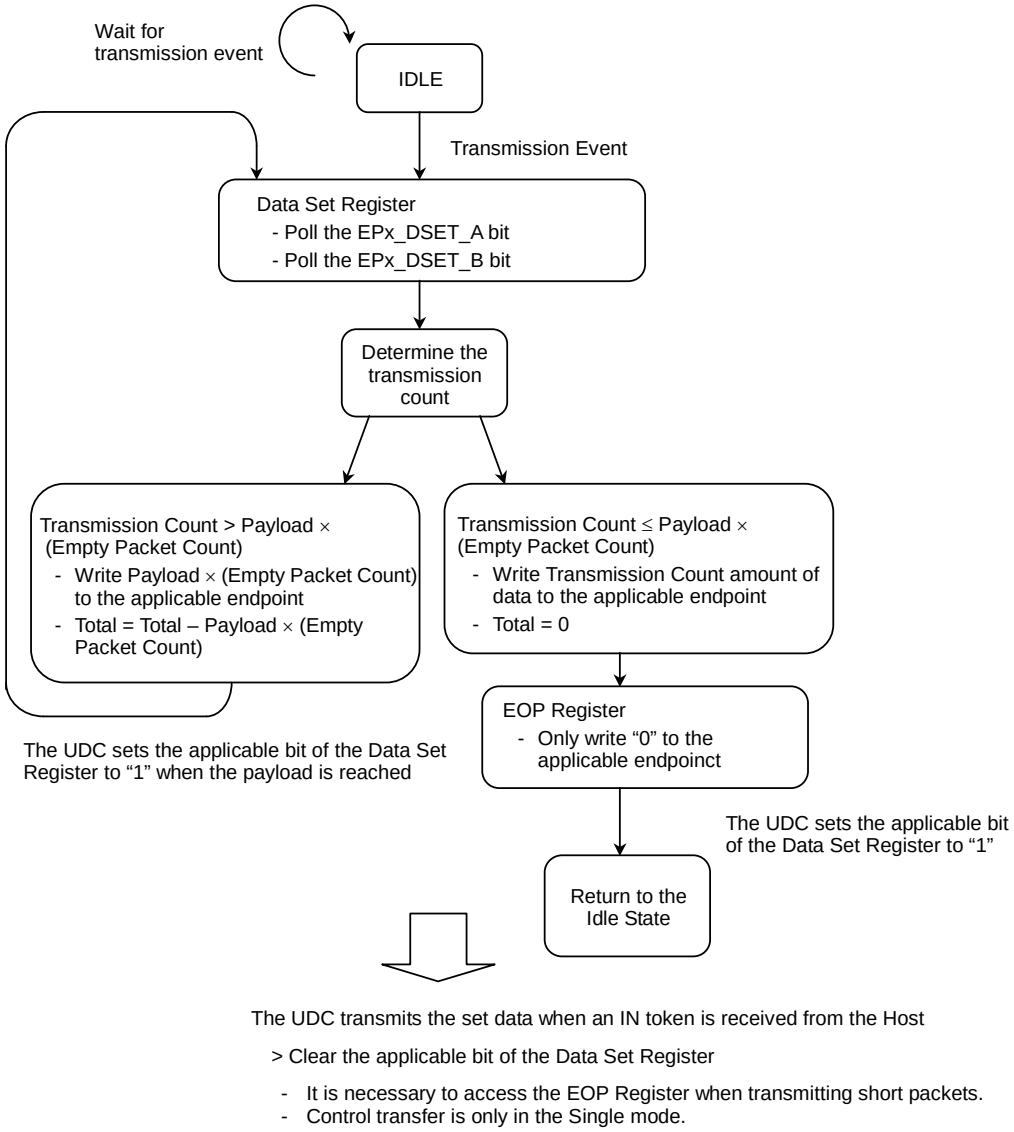


Figure 8.1.11 Transmission Sequence in the Dual Packet Mode

8.1.5.2 DMA Bus Interface

A total of seven DMA Bus Interface sets are available for endpoints 1 – 2.

- EPx_BCS: This bit specifies how an endpoint is accessed.

  0: DMA access

  1: CPU access

  Please refer to 8.2.3.3 Interrupt Cause Enable Register (0x04-0x07).

- EPx_BRD: This is an active-low input used to read data from a Endpoint.

- EPx_BWR: This is an active-low input used to write data to a Endpoint.

- EPx_EOPB: A negative-going pulse on this bit indicates completion of a short-packet transfer.

- EPx_DATASET:

  The content in the Dataset Register (8.2.3.28 Dataset Register (0x330)) is driven to this output to indicate whether data is present in a Endpoint. A "high" indicates that there are data sets in a Endpoint. A "low" indicates that the Endpoint is empty. Hence, a FIFO is ready to accept data when it is at logic low during transmit operations and at logic high during receive operations. The Dataset Register have two bits representing packets A and B for each endpoint. The EPx_DATASET bit (refer to 8.2.3.2 Interrupt Cause Status Register (0x00-0x03)) goes high when at least either one of these bits is set to 1. Applications can monitor this signal as a DMA data request signal whether in single packet mode or dual packet mode.

- EPx_DATA[7:0]:

  These signals provide a bidirectional data Bus between the UDC and an DMA controller.

8.1.5.2.1 DMA transfer in the Reception mode

When the reception transfer of one packet of data is complete and the data is prepared in Endpoint, the EPx_DATASET bit (D6 – 8 of 8.2.3.2 Interrupt Cause Status Register (0x00-0x03)) is set to the "H" level and receipt of the data is signaled outside the UDC. If DMA setting is finished, DMA transfer automatically starts.

The following figure shows example timing when in the Dual Packet mode (with an 8-byte payload).

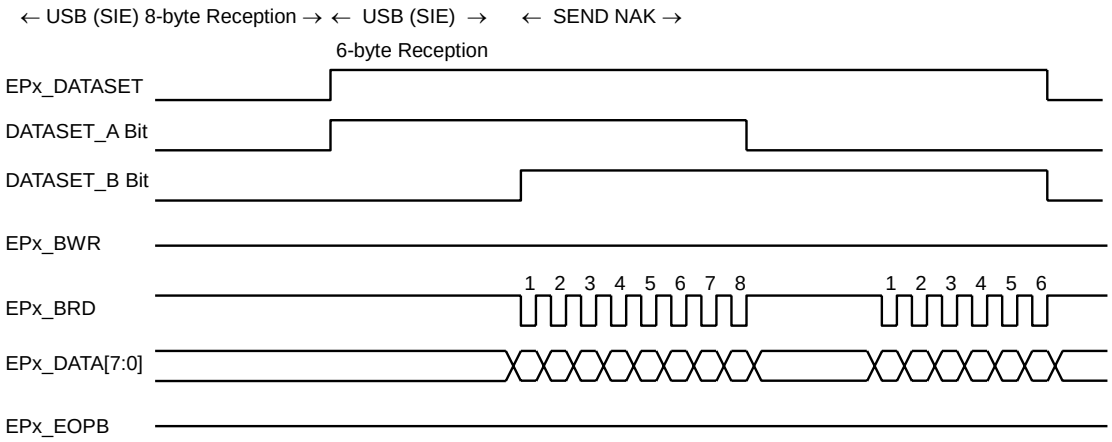


Figure 8.1.12 DMA I/F Timing in the Reception Mode

8.1.5.2.2 DMA transfer in the Transmission mode

When in the Transmission mode, the outside of the UDC is signaled that writing is enabled if the EPx_DATASET bit(D6 – 8 of 8.2.3.2 Interrupt Cause Status Register (0x00-0x03)) is at the "L" level. The DMA circuit of GOKU-S responds to this bit being at the "L" level by starting a write transaction. When in the Dual Packet mode, the UDC sets the EPx_DATASET bit to the "H" level when two packets of data are transferred to the Endpoint in order to signal that writes to the Endpoint are disabled.

The UDC responds to a valid IN token for this endpoint and transmits data when one packet of valid data is ready. There is empty space in the Endpoint even when one packet of data is completely transmitted, so the UDC returns EPx_DATASET bit to the "L" level and sends write permission to the outside of the UDC.

The UDC automatically transmits data at the maximum payload size for each endpoint. When transmitting short packet data however, it is set the MSTrd_eopb bit(D7 of Master Operation Setting Register) to "1" by software when writing of the short packet size data is complete. The UDC then judges that the packet has ended and transmits the short packet.

The following figure shows example timing when in the Dual Packet mode (with an 8-byte payload).

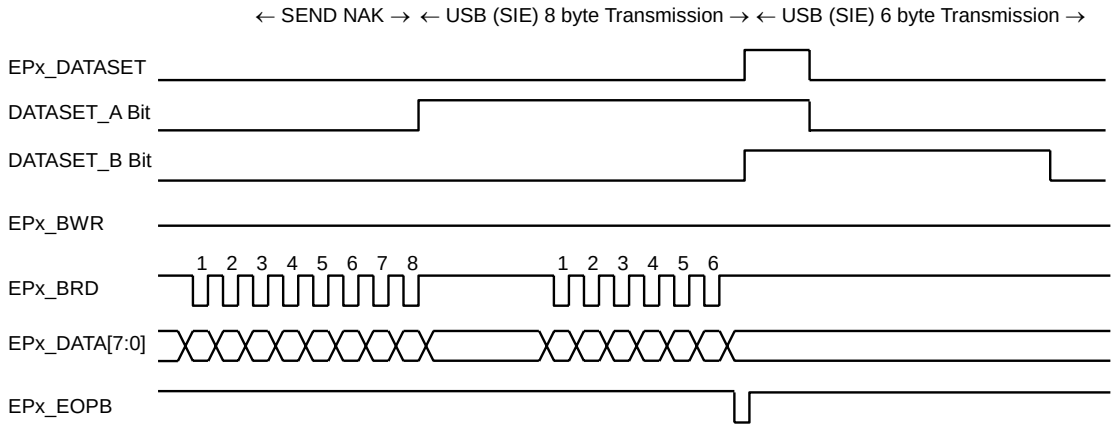


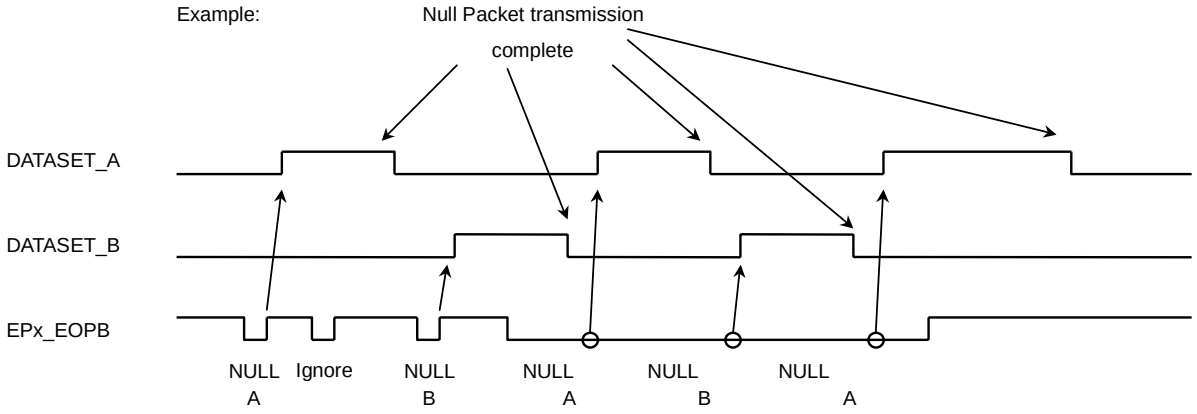
Figure 8.1.13 DMA I/F Timing in the Transmission Mode

8.1.5.2.3 Issuing Null packets

When transmitting Null packets, inputting an "L" pulse from the EPx_EOPB signal sets data with a length of 0 in the ENDPOINT and makes it possible to transmit a NULL packet to the IN token. However, Null data can only be set in the ENDPOINT when the DATASET signal is in the "L" level state (namely, when there is no data in the ENDPOINT). If all tokens received within a certain period are responded to with Null packets, then keeping the EPx_EOPB signal at the "L" level makes it possible to respond.

A Null packet can also be set by accessing the EOP register.

When in the Dual Packet mode however, the "L" level is asserted so the EPx_DATASET signal can indicate data space. Therefore, it is not possible to know from outside the UDC if there is no data in either bank.



8.1.5.3 Interrupts

USB Device Controller interrupts consist of interrupts that are generated from the USB Device Core (UDC) and interrupts that are generated from other sources. When the interrupt conditions are met, the USB Device Controller sets the corresponding bit of the internal Interrupt Status Register. A hardware interrupt is generated if the applicable bit of the Interrupt Enable Register is set to Enable when this bit is set.

"1" is asserted to the corresponding Status Register bit if an interrupt cause occurs when the Interrupt Enable Register is set to Disable. A hardware interrupt will not be generated however. If the Interrupt Enable bit is set to Enable when the Interrupt Status Register is asserted, a hardware interrupt will be generated immediately after the bit is set.

The default for all Interrupt Enable Register bits except for the INT_STATUSNAK bit is "0".

The interrupts that are generated from the USB Device Core correspond to bits 0 – 13 of the Interrupt Status Register. The USB Device Core is designed to use these interrupts to realize a software response to Device requests sent from the USB Host. Please mask any unnecessary interrupt causes when performing hardware responses. Please refer to interrupt signals (8.4.1) that are generated from the USB Device Core.

8.1.5.3.1 Interrupt control

Following is a description of the available interrupt signals. Please use them in accordance to the system specifications.

- INT_SETUP

  After receiving a Setup packet, first it is judged whether the received request should be automatically responded to or is a request that requires control authority to be transferred to the software. This interrupt is only asserted to the outside when the received request requires software control. When this interrupt is received, the software reads out an 8-byte Device request from a register in the UDC. Please perform the process that is appropriate for each request.

- INT_ENDPOINT0

  This interrupt is asserted when a Data packet is received or transmitted in response to a request that has a Data stage. It is necessary to receive data from the ENDPOINT when this interrupt is received in a Control Write transfer. It is necessary to write new data that will be transmitted to the ENDPOINT when this interrupt is accepted in a Control Read transfer. In addition, there are cases in which this interrupt cannot be asserted since some Host Controller types may not have an ACK in the last packet of the Data stage. The amount of data that must be transmitted is either specified from the Host or varies according to the data size on the device side. Therefore, please be sure that this interrupt is ignored if it is generated after data for the last packet of the Data stage is written.

- INT_STATUS

  This interrupt signal is asserted when the Status stage ends. The request ends normally if this interrupt is received. If a Setup packet is received without this interrupt signal, then the previous request did not end normally and this interrupt is displayed in the EP0_Status Register as STAGE_ERR.

- INT_STATUSNAK

  This interrupt becomes necessary in the case of a Control Read transfer when the Status stage is shifted to at a value that is less than the data length (wLength) specified from the Host. A NAK to the first packet of the Status stage is used and an interrupt signal is created. This interrupt is disabled when a Setup packet is received, so please enable it using the Interrupt Control Register when it is to be used.

- EP1, 2, 3_DATASET

  This is the Data request signal for endpoints 1 – 3. The logic varies depending on whether data is being transmitted or received, so controlling the polarity makes it possible to also use this signal as an interrupt.

- INT_EP1, 2, 3NAK

  This interrupt is available for all endpoints except for endpoint 0. This interrupt outputs a High pulse for one USB_CLK clock cycle when a NAK was returned to the USB Host from the applicable endpoint. Please process this interrupt using external logic if it is to be used when disabled since it is not synchronized to the ENDPOINT disable function.

- CLASS_REQUEST

  This output signal goes high when the UDC has received a class request and is held there until the EOP register is accessed.

- VENDOR_REQUEST

  This output signal goes high when the UDC has received a vendor request and is held there until the EOP register is accessed.

- USB_RESET

  Only output the “H” level while a USB reset is in progress. The application must be initialized using this reset.

8.1.6 USB Device Response

The UDC sets the initialization inside the UDC and sets each register for when a Hardware reset is detected, a USB Bus reset is detected, or when there is an enumeration response. Each state is described below.

8.1.6.1 Hardware reset is detected

The UDC must always input a Hardware reset signal after the power is turned on. The UDC initializes the Status register as follows when it detects a Hardware reset from the outside.

Register Name	Initial Value
EPx_Status	0x1C

All endpoints are disabled when the UDC is in this state. Signals from the USB Host only accept USB Reset signals.

8.1.6.2 Bus reset is detected

The UDC initializes internal registers and prepares for enumeration operation from the USB Host when it detects a Bus reset in the USB signal line. The UDC sets endpoint 0 to Control transfer, 8-byte payload, and sets the default address to make it possible to use the default type after a USB reset is detected. The UDC sets other endpoints to the disabled state.

Register Name		Initial Value
Endpoint Status	EP0:	0x00
	EP1-3:	0x1C

8.1.6.3 Status Register details

The Status Register available for each endpoint indicates the status of each UDC endpoint. Each status affects each type of USB transfer. Please refer to Section 8.1.4 for information regarding the status changing during each type of transfer.

The value of the Endpoint Status Register is 0 – 7. Each value expresses the states shown below. Displays 0 – 4 indicate the results of each transfer type. Confirming from outside the UDC makes it possible to confirm the immediately previous transfer result of the endpoint.

0: Ready
1: Data In
2: Full
3: TX_ERR
4: RX_ERR

These states indicate that the applicable endpoint is operating normally. The meaning of the display varies according to the transfer mode, so please refer to the following transfer mode columns.

Transfer mode other than ISO

Indicates the results of the immediately previous transfer. It is updated when transfer is complete.

	OUT, Setup	IN
Initialization	Ready	Ready
Normal transfer completion	Data In	Ready
Status stage completion	Ready	Ready
Error transfer	RXERR	TXERR

The term “initialization” above refers to when there is a Reset, USB Reset, or the CurrentConfig Register is updated. When an error is detected, EPx_DATASET is generated except for when in the Interrupt Toggle transfer mode or Isochronous transfer mode.

Status Register displays 5 – 7 indicate that the endpoint is in a special state.

5: Busy Is only generated at endpoints that perform Control transfer. This bit is set if a Status stage ID is received from the USB Host when the CPU has not completed the enumeration process and the UDC is still performing a Control Write transfer. The status remains Busy until the CPU completes the enumeration process and “0” is written to the EP0 bit of the EOP Register in the UDC. The Ready status is displayed if the enumeration process ends, “0” is written to the EP0 bit of the EOP Register, and the Status stage from the USB Host ends normally. Please refer to 8.1.4.4 for more information.

6: Stall Indicates that an endpoint is in the Stall state. This status occurs either when a protocol violation occurred or if an error occurred during Bus enumeration. A Device request by the USB Host is required in order to return the endpoint to a state where normal transfer is possible.

7: invalid This status indicates that the endpoint is in a state where it cannot be used. The UDC sets the endpoint that is not specified by the Descriptor to the Invalid state and ignores all tokens for that endpoint. This status always occurs during initialization. The UDC sets all endpoints to the Invalid state when it detects a Hardware reset. Then, it only updates endpoint 0 to the Ready state when a USB Reset is received. Other endpoints defined in the descriptor are updated to the Ready state when the SET_CONFIG request is completed normally.

8.1.7 Power Management

The USB Device Controller (UDC) can change from any Resume state (Power supply On state) to the Suspend state, or can recover the power supply On state from the Suspend state. Operating the CLK supplied to the UDC from outside the UDC makes it possible to further reduce power consumption.

8.1.7.1 Change from the Suspend state

The USB Host can set a USB device to the Suspend state by maintaining the Idle state. The UDC changes to the Suspend state in the following order.

- The UDC changes to the Suspend state when it detects an Idle state on the USB signal line that continued for 3 ms or more. The UDC sets the Suspend bit of the Status Register to "1" at this time.
- When 2 ms passes after changing to the Suspend state, the UDC updates the output of the Suspend output pin from "0" to "1". Operation of the CLK (USB_CLK) inside the UDC is suspended at this time.
- All values in the registers inside the UDC are retained at this time, but the only permitted access from outside the UDC to read the Status Register and Current Config Register.

8.1.7.2 Recovering from the Suspend state by Host Resume

The UDC resets the Suspend output from "1" to "0", resets the Suspend bit of the Status Register to "0", and causes system operation to resume. This is done when the Resume status output from the USB Host recovers bus activity on a USB signal line. Since the Resume status output from this Host is maintained for at least 20 ms, valid protocols will be generated on a USB signal line after this time elapses.

8.1.7.3 Recovering from the Suspend state by Remote Wakeup

GOKU-S does not support Remote Wakeup.

8.1.8 RESET

The USB Device Controller supports initialization by hardware reset (FULLRST*) and power reset by setting the Power Reset bit (software reset). Always be sure to assert a hardware reset and initialize all registers after turning the power On.

It is also necessary to assert a reset when USB Bus power is detected. In this case, please be sure to use the Power Reset bit of the Power Supply Detection Register to perform a software reset.

Table 8.1.1 below shows the reset and default for each USB Device Controller register. Registers with the letter "H" in the Reset column are registers that can only be initialized by hardware reset input. Registers with the letters "H/S" in the Reset column can be initialized by hardware reset input and software resets. The default items during reset are as follows below.

Table 8.1.1 USB Device Controller Register: Resets and Defaults

Address	Register Name	Bit	Mnemonic	Reset	Default
+0x024	Power Supply Detect Control Register	2	PW_DETECT	H	0
		1	PW_ResetB	H	1
		0	PULLUPENB	H	0
+0x020	Master Read Current Address Register	31:00	MSTrd_crt_address_REG	H/S	0xFFFF_FFFF
+0x01c	Master Read End Address Set Register	31:00	MSTrd_end_address_REG	H/S	0xFFFF_FFFF
+0x018	Master Read Start Address Set Register	31:00	MSTrd_start_address_REG	H/S	0xFFFF_FFFF
+0x014	Master Write Current Address Register	31:00	MSTwr_crt_address_REG	H/S	0xFFFF_FFFF
+0x010	Master Write End Address Set Register	31:00	MSTwr_end_address_REG	H/S	0xFFFF_FFFF
+0x00c	Master Write Start Address Set Register	31:00	MSTwr_start_address_REG	H/S	0xFFFF_FFFF
+0x008	Master Operation Setting Register	11	eopb_disable	H/S	0
		10	eopb_enable	H/S	1
		9	Timeout_disable	H/S	0
		8	Timeout_enable	H/S	1
		7	MSTrd_eopb	H/S	0
		6	MSTrd_Reset	H/S	0
		5	MSTwr_Reset	H/S	0
		2	MSTrd_enable	H/S	0
		1	MSTwr_enable	H/S	0
		0	MST_connection	H	0

Address	Register Name	Bit	Mnemonic	Reset	Default
+0x004	Interrupt Cause Enable Register	19	INT_PowerDetect	H	0
		18	INT_system_error	H	0
		17	MSTrd_end_add	H	0
		16	MSTwr_Timeout	H	0
		15	MSTwr_end_add	H	0
		14	MSTwr_set_add	H	0
		13	ERR	H	0
		12	SOF	H	0
		11	INT_EP3NAK	H	0
		10	INT_EP2NAK	H	0
		9	INT_EP1NAK	H	0
		8	EP3_DATASET	H	0
		7	EP2_DATASET	H	0
		6	EP1_DATASET	H	0
		5	INT_STATUSNAK	H	1
		4	INT_STATUS	H	0
		3	INT_SETUP	H	0
		2	INT_ENDPOINT0	H	0
		1	USB_RESET	H	0
		0	SuspendResume	H	0
+0x000	Interrupt Cause Status Register	19	INT_PWdetect	H	0
		18	INT_syserr	H/S	0
		17	MSTrd_end_add	H/S	0
		16	MSTwr_Timeout	H/S	0
		15	MSTwr_end_add	H/S	0
		14	MSTwr_set_add	H/S	0
		13	ERR	H/S	0
		12	SOF	H/S	0
		11	INT_EP3NAK	H/S	0
		10	INT_EP2NAK	H/S	0
		9	INT_EP1NAK	H/S	0
		8	EP3_DATASET	H/S	0
		7	EP2_DATASET	H/S	0
		6	EP1_DATASET	H/S	0
		5	INT_STATUSNAK	H/S	0
		4	INT_STATUS	H/S	0
		3	INT_SETUP	H/S	0
		2	INT_ENDPOINT0	H/S	0
		1	USB_RESET	H/S	0
		0	SuspendResume	H/S	0

Note about Master Channels:

Each Master Channel (Read/Write) has a Master Channel Reset. All these resets do is to reset the applicable Master Channel. Note that these resets **do not** reset the USB Device Controller Register. Please refer to Section 8.2.3.4 Master Operation Setting Register (0x08-0x0B) for more information regarding the method of resetting each Master Channel.

8.1.9 Operation Sequence

The USB Device Controller operation sequence is as follows below.

(1) Hardware Reset

This reset initializes all registers. Always assert a hardware reset after turning the power On.

(2) Set operation mode

Set the Master Connection setting and Interrupt Cause Enable of the Master Operation Setting Register. Please refer to Sections 8.2.3.3 Interrupt Cause Enable Register (0x04-0x07) and 8.2.3.4 Master Operation Setting Register (0x08-0x0B) for more information.

(3) Detect USB Bus Power Supply

Please refer to Section 8.1.10 Power Supply Detection Sequence for more information.

(4) USB Device Controller Setup

After making settings such as the UDC Register, Descriptor RAM, and the USB Device Controller Register, use the USBREADY Register to signal to the UDC that data writing is complete. Pulling up the USB Data Line after that transitions to Enumeration Response. Please refer to Section 8.1.10 of this document.

(5) USB Enumeration Response

(6) a. Master Write Transfer

Master Write transfer is performed if a Transmission request is generated from the USB Host Controller. Please refer to Section 8.1.11.2 Master Write Transfer for more information.

b. Master Read Transfer

Master Read transfer is performed if a Reception request is generated from the USB Host Controller. Please refer to Section 8.1.11.1 Master Read Transfer for more information.

(7) Disconnect

If the USB Power Supply is not detected, the process will return to this status and an interrupt will be generated regardless of which status of 4 – 6 above is the current status. Please set a software reset from the Power Supply Detection Register.

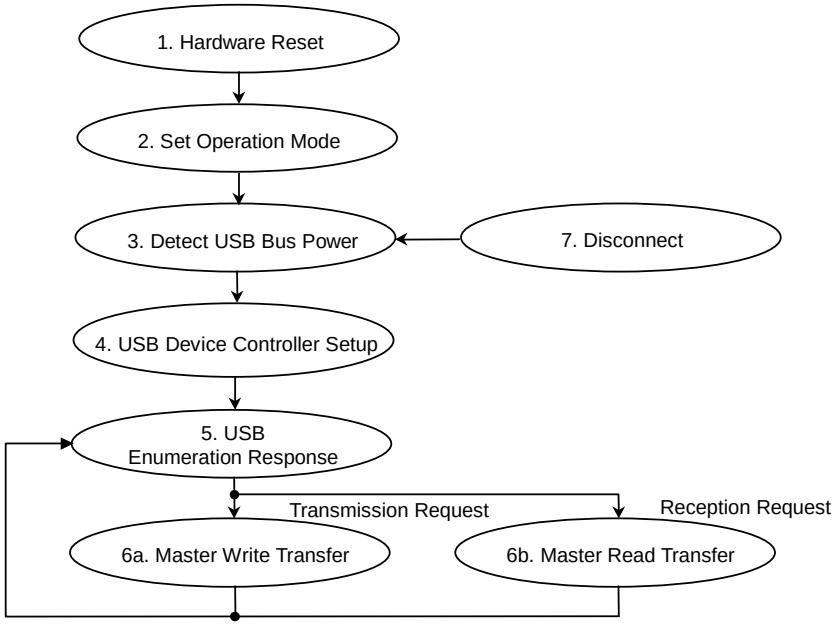


Figure 8.1.14 Operation Sequence

8.1.10 Power Supply Detection Sequence

The USB Device Controller must detect the USB Bus power supply and perform initialization. At this time, the UDC clears the software reset (PW_ResetB*), then disables detection of the USB_RESET signal until “0” is written to the USBREADY Register.

This section describes the sequence pertaining to power supply detection.

If power supply detection interrupts are enabled by the software after a hardware reset is performed, then a hardware interrupt (INT_PowerDetect) is generated when the power supply is detected. Please use the software to assert (more than 250 μsec) a software reset (PW_ResetB) if a hardware interrupt is detected. Please use the software to deassert PW_ResetB since it is not automatically cleared. The USB Device Controller is now in the setup state. Please use the software to set the UDC Register, Descriptor RAM, and the USB Device Controller Register. After setting is complete, end setup by setting “0” in the USBREADY Register inside the UDC. Finally, enabling Pull up of the USB Data line transitions the UDC to the enumeration process.

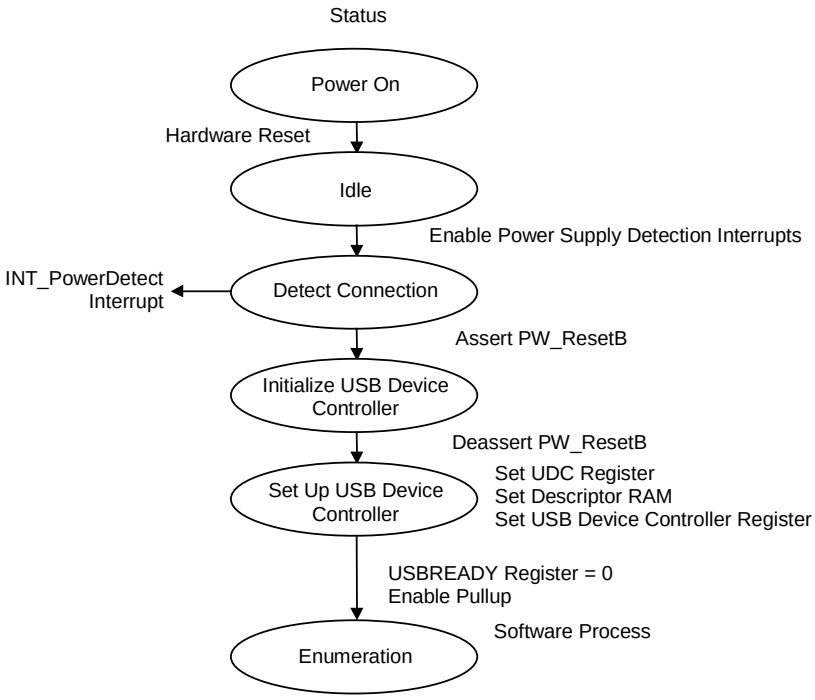


Figure 8.1.15 Power Supply Detection Sequence

8.1.11 Master Transfer Operation

This section describes the Master Transfer operation of the USB Device Controller.

8.1.11.1 Master Read Transfer

8.1.11.1.1 EOP Enable Mode

This section describes Master Read transfers when Master Read EOP is enabled. Master Read operation is performed according to the following operation.

(1) Please complete set up as far as USB Device Controller enumeration.

(2) The software sets the Master Read Start Address and Master Read End Address.

(3) Next, “1” is set to the Master Read/Write Enable bit of the Master Operation Setting Register.

(4) The USB Device Controller begins transferring data to the UDC endpoint. The UDC transfers the In token from the USB Host then transfers data.

(5) The Master Read block asserts the EOP signal to the UDC and sets the MSTrd_end_add interrupt when the Master Read End Address is reached. When an IN token is designated from the USB Host, this operation transmits either a short packet or a null packet one time.

(6) The USB Device Controller returns to step 2 above if the software response is already complete.

There are no timeouts in the Master Read Block.

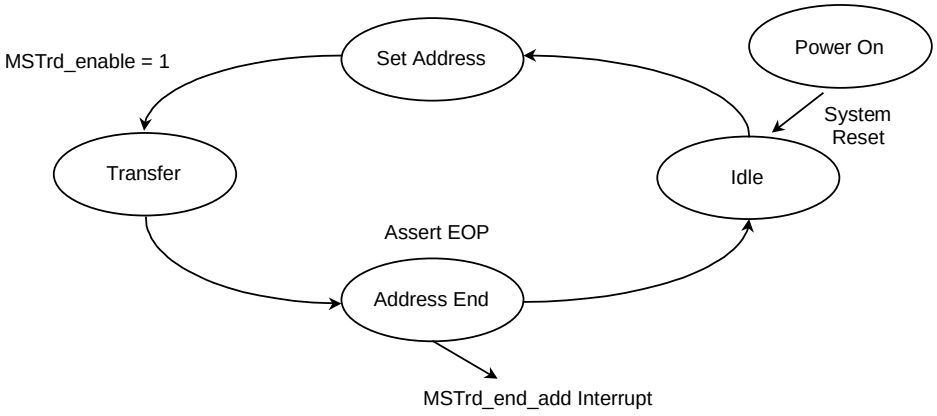


Figure 8.1.16 Master Read Sequence (Enable EOP Mode)

Note about the MSTrd_end_add interrupt:

The MSTrd_end_add interrupt is generated when data transfer to the UDC endpoint ends. Please refer to the UDC DATASET Register to confirm whether all data transfers to the USB Host are complete.

8.1.11.1.2 EOP Disable Mode

This section describes Master Read transfers when Master Read EOP is disabled. Master Read operation is performed according to the following operation.

(1) Please complete set up as far as BUSBDC enumeration.

(2) The software sets the Master Read Start Address and Master Read End Address.

(3) Next, "1" is set to the Master Read/Write Enable bit of the Master Operation Setting Register.

(4) The USB DEVICE CONTROLLER begins transferring data to the UDC endpoint. The UDC transfers the IN token from the USB Host then transfers data.

(5) The USB DEVICE CONTROLLER sets the MSTrd_end_add interrupt when the Master Read End Address is reached. Data is transferred after the IN token from the USB Host if the FIFO Register of the endpoint reaches the payload during Master Read transfer. However, the data remains in the FIFO Register and is held for transfer to the next endpoint if the payload is not reached.

(6) Operation returns to step 2 above if the software response is already complete.

There are no timeouts in the Master Read Block.

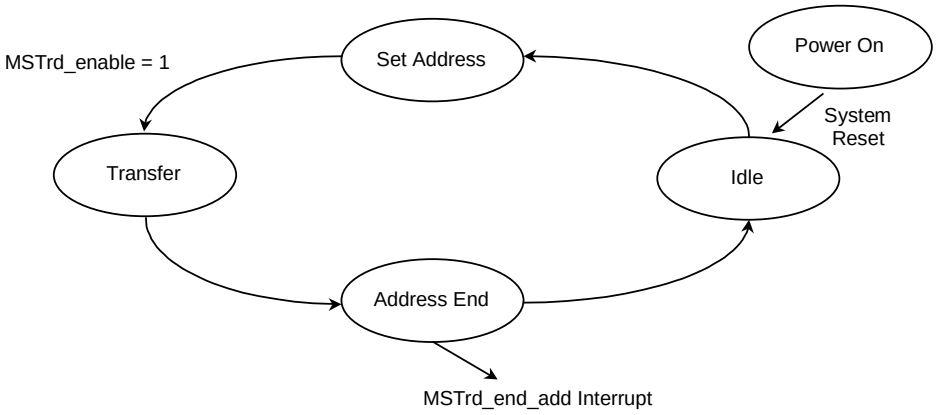


Figure 8.1.17 Master Read Sequence (Disable EOP Mode)

Note about short and null packets:

When using the USB DEVICE CONTROLLER in the Master Read EOP mode, neither short packets nor null packets is transmitted even if transfer of the data string to be transmitted is complete. Please check the MSTrd_end_add interrupt. If transfer of the data string to be transmitted is complete, please use the Master Read EOP bit of the Master Operation Setting Register to transmit either a short packet or a null packet.

However, both DATASET_A and DATASET_B of the applicable endpoint must be set to "0" in order to transmit a null packet. Please use the DATASET Register of the UDC to confirm the value when transmitting a null packet.

8.1.11.2 Master Write Transfer

This section describes the Master Write operation. Master Write operation is performed according to the following operation.

(1) Please complete set up as far as USB Device Controller enumeration.

(2) The USB Device Controller generates an MSTwr_set_add interrupt when it receives an error-free packet from the USB Host.

(3) Please use the software to set the Master Write Start Address and the Master Write End Address.

(4) Set "1" to the Master Write Enable bit of the Master Operation Setting Register.

(5) The USB Device Controller performs Master Write transfer on the data received from the USB Host.

(6) If reception of a packet series is interrupted and an SOF is received three times before the current address reaches the end address, the USB Device Controller terminates the Master process and generates an MSTwr_Timeout interrupt. Please use the software to perform a termination process. After this interrupt is generated, operation returns to step 2 above when the UDC receives an error-free packet.

(7) An MSTwr_end_add interrupt is generated if writing is complete up to the Master Write End Address when a Timeout interrupt has been generated. Please use the software to run a termination process. After this interrupt is generated, operation returns to step 2 above when the UDC receives an error-free packet.

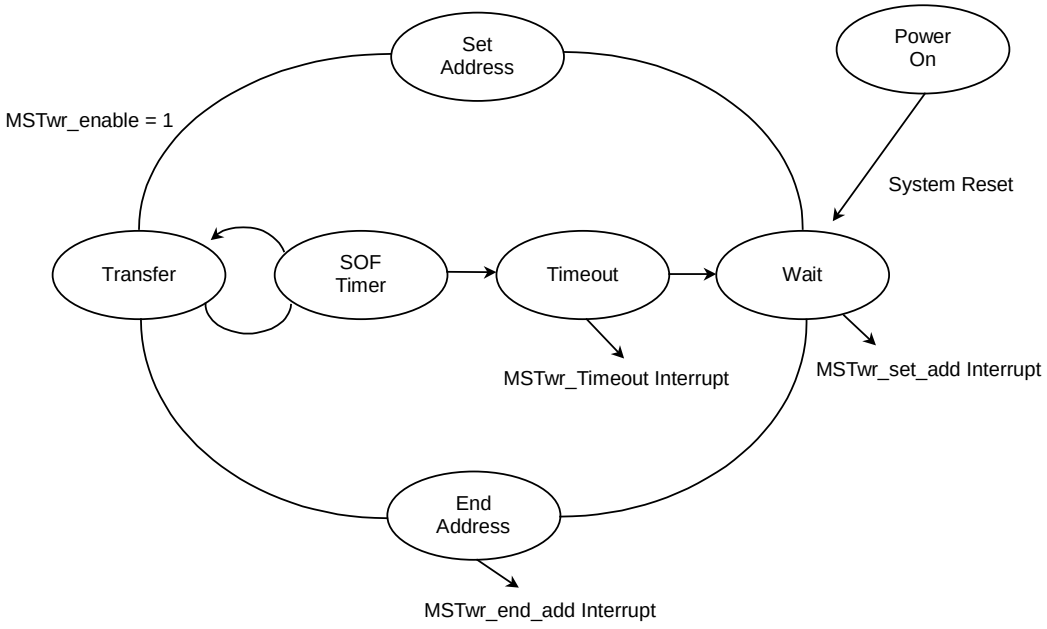


Figure 8.1.18 Master Write Sequence

Note about Timeout process:

When using the USB DEVICE CONTROLLER to perform Master Write transfer, there are cases in which Timeout interrupts occur. The UDC that is built into the USB DEVICE CONTROLLER performs error checks on packets transferred from the USB Host and stores data in the internal FIFO. The Master Write Block fetches data from the UDC in 8-bit increments and stores the data in 4-word Transfer FIFO. When four words of data are stored in the FIFO, the Master Write Block performs Master Write transfer with memory.

The USB protocol is based on specifications that make it possible to transmit data of any size that is less than the Max packet size. If the data fetched from the UDC is less than four words, then the USB DEVICE CONTROLLER waits for the next packet. If an SOF packet is received three times in this situation, then the USB DEVICE CONTROLLER performs Master transfer on data that is in the Transfer FIFO in the Master Write Block. After that, the USB DEVICE CONTROLLER sets a timeout interrupt and clears MSTwr_enable in the Master Operation Setting Register.

After this interrupt is detected, then the software can confirm the transfer destination address of the final data in the current address register. It is necessary to do so from the setting of the Master Write Start Address when a new packet is received again. However, when setting up transfer after the previous data string, please set the next byte value after the last data address to the Master Write Start Address.

It is now possible to use the Timeout Disable function of the Master Operation Setting Register to disable the previously mentioned Timeout function. Transfer will not end until the defined end address is reached.

Note about disabling Master Write transfer:

After making all necessary settings and after Master Write transfer starts, it is possible to use the Master Write Disable bit of the Master Operation Setting Register to suspend Master Write transfers.

In this situation, the USB DEVICE CONTROLLER immediately terminates transfer from the UDC endpoint to the FIFO used for Master Write transfer. After that, Master transfer is performed to transfer the data in the FIFO used for Master Write transfers to memory, then the Timeout interrupt is set and Master Write transfer ends.

Please use the Master Write Current Address Register to confirm the address at which transfer ended.

8.1.11.3 Stall Process

The USB DEVICE CONTROLLER does not assume that endpoints other than Endpoint 0 will be in the Stall state. Therefore, operation may be performed out of sync with Master transfer if the UDC automatically replies to a Clear Feature request after the endpoint has stalled. It will be necessary to initialize Master transfer in order to synchronize to Master transfer. However, it will be difficult to do so by just using the software when using the Auto Reply function. It is therefore preferable to manually reply to Clear Feature requests when using applications with which endpoints other than Endpoint 0 stall.

8.2 Register Description

The USB Device Controller has one PCI configuration space and USB Device Controller's memory addresses. The memory registers are accessed after setting the base address in the configuration register.

Table 8.2.1 Register Access Type

Access Type	Meaning
RO	Read-only
WO	Write-only
RW	Can both be read and written to.
RWC	Can both be read and written to. Cleared when written to.

8.2.1 Configuration Map

23		15	07	00
Device ID			Vendor ID	0x00
Status			Command	0x04
Class Code			Revision ID	0x08
BIST	Header Type	Latency Timer	Cache Line Size	0x0C
UDC memory base address				0x10
				0x14
				0x18
				0x1C
				0x20
				0x24
				0x28
Sub System ID		Sub Vendor ID		0x2C
				0x30
			Capability Pointer	0x34
				0x38
Maximum Latency	Minimum GNT	PCI Interrupt Pin	PCI Interrupt Line	0x3C
		Retry Timeout	TRDY Timeout	0x40
			PCI Interrupt Pin Select	0x44
				0x48-0xD8
Power Management Capabilities		Next Item Pointer	Capability ID	0xDC
Data	PMCSR BSE	Power Management Control/Status		0xE0
				0xE4-0xFC

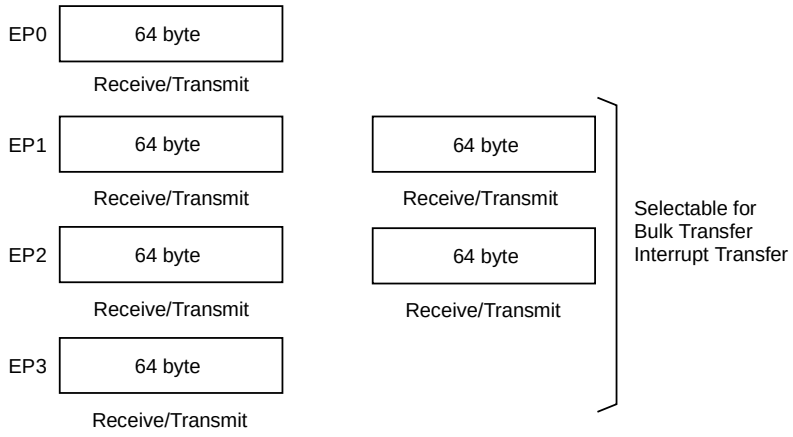
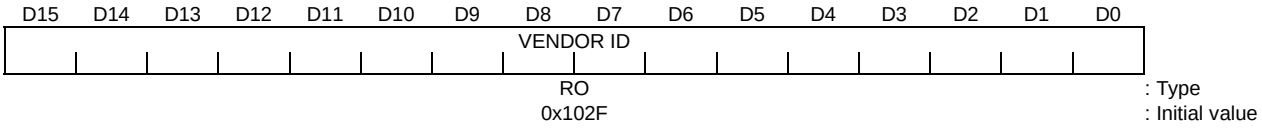


Figure 8.2.1 USB Device FIFOs

8.2.2 Configuration Register Details

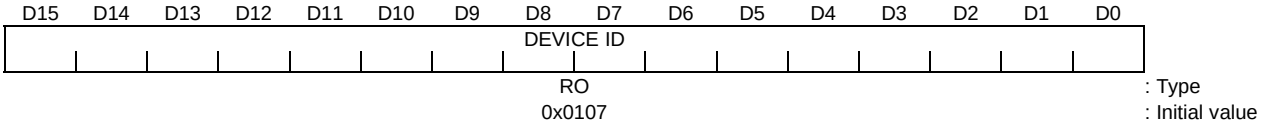
8.2.2.1 Vendor ID Register (0x00–0x01)



Bits	Mnemonic	Field Name	Description
D15:D0	VENDOR ID	Vendor ID	Vendor ID This value 0x102F represents the Toshiba Semiconductor Company.

Figure 8.2.2 Vendor ID Register

8.2.2.2 Device ID Register (0x02–0x03)



Bits	Mnemonic	Field Name	Description
D15:D0	DEVICE ID	Device ID	Device ID This value 0x0107 represents the USB Device Controller.

Figure 8.2.3 Device ID Register

8.2.2.3 Command Register (0x04–0x05)

D15	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	
Reserved		FBBE	SER	WCC	PER	VGA	MWI	SP	BM	MS	IOS	
		RW	RW	RO	RW	RO	RW	RO	RW	RW	RO	: Type
		0	0	0	0	0	0	0	0	0	0	: Initial value

Bits	Mnemonic	Field Name	Description
D15:D10		Reserved	
D9	FBBE	Fast Back-to-Back Enable	Fast Back-to-Back Enable This bit controls whether or not a master can do fast back-to-back transactions to different devices. A value of 1 means the master is allowed to generate fast back-to-back transactions to different agents. A value of 0 means fast back-to-back transactions are only allowed to the same agent.
D8	SER	SERR Enable	SERR Enable This is the enable bit for the SERR* driver. All devices that have an SERR* pin must implement this bit, also bit[6] must be on to report address parity errors when SERR* is “1”. 1: Device must take its normal action when a parity error id detected. 0: Device must set its parity errors detection status bit (bit[15] in the status register) when an error occurred but does not assert PERR* and continues its normal operation (Default).
D7	WCC	Wait Cycle Control	Wait Cycle Control Fixed to “0” since the address/data are not stepped.
D6	PER	Parity Error Response	Parity Error Response This bit controls the device’s response to Parity Error. 1: Enable SERR* driver 0: Disable SERR* driver (Default)
D5	VGA	VGA Palette Snoop	VGA Palette Snoop Fixed to “0” since this module is not a VGA device.
D4	MWI	Memory Write and Invalidate Enable	Memory Write and Invalidate Enable
D3	SP	Special Cycle	Special Cycle Fixed to “0” since this module does not respond to special cycles.
D2	BM	Bus Master	Bus Master Set this bit to “1” to operate this module as a PCI bus master.
D1	MS	Memory Space	Memory Space Set this bit to “1” to support UDC memory registers.
D0	IOS	I/O Space	I/O Space Fixed to “0” since this module does not have I/O registers.

Figure 8.2.4 Command Register

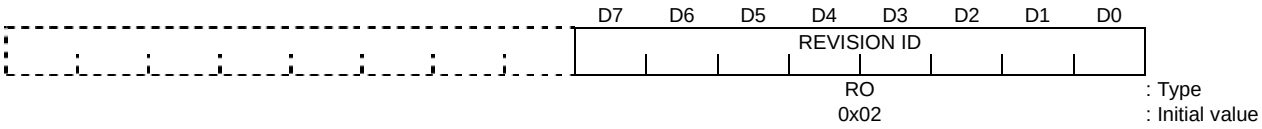
8.2.2.4 Status Register (0x06–0x07)

D15	D14	D13	D12	D11	D10 D9	D8	D7	D6 D5	D4	D3 D0	
DPE	SSE	RMA	RTA	STA	DECT	DPD	FBBC	Reserved	CAP	Reserved	
RWC	RWC	RWC	RWC	RWC	RO	RWC	RO	RO	RO	RO	: Type
0	0	0	0	0	01	0	1	00	1	0x00	: Initial value

Bits	Mnemonic	Field Name	Description
D15	DPE	Detected Parity Error	Detected Parity Error This bit must be set by the device whenever it detects a parity error, even if parity error handling is disabled (as controlled by bit[6] in the Command register bit). 1: Reset 0: No action (Default)
D14	SSE	Signaled System Error	Signaled System Error This bit must be set whenever the device asserts SERR*. Devices who will never assert SERR* do not need to implement this bit. 1: SERR* asserted 0: No action (Default)
D13	RMA	Received Master Abort	Received Master Abort Is set to “1” when PCI master access performed by this module ends with a master abort.
D12	RTA	Received Target Abort	Received Target Abort Is set to “1” when PCI master access performed by this module ends with a target abort.
D11	STA	Signaled Target Abort	Signaled Target Abort Is set to “1” when a target abort occurs while this module is the PCI target.
D10:D9	DECT	DEVSEL Timing	DEVSEL Timing Specifies DEVSEL response timing. These bits are fixed to “01b” since this module responds using the medium DEVSEL timing.
D8	DPD	Master Data Parity Detected	Master Data Parity Detected This bit is only implemented by Bus master. It is set when three conditions are met: 1). The bus agent asserted PERR* itself on a read or observed PERR* asserted on a write; 2). The agent setting the bit acted as the bus master for the operation in which the error occurred; and 3). The Parity Error Respond bit (Command Register’s bit[6]) is set 1: Reset 0: No action (Default)
D7	FBBC	Fast Back-to-Back Capable	Fast Back-to-Back Capable This bit indicates whether or not target is capable of accepting fast back-to-back transactions when the transactions are not to the same agent. This bit is fixed to “1” since the device can accept these transactions.
D6:D5		Reserved	
D4	CAP	Capabilities	Capabilities Set this bit to “1” to support PCI Power Management. This function is supported by default.
D3:D0		Reserved	

Figure 8.2.5 Status Register

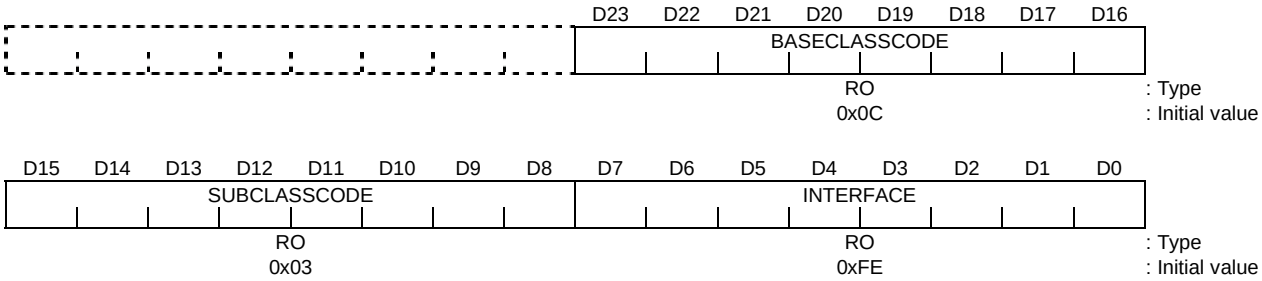
8.2.2.5 Revision ID Register (0x08)



Bits	Mnemonic	Field Name	Description
D7:D0	REVISION ID	Revision ID	Revision ID This value 0x02 represents the revision.

Figure 8.2.6 Revision ID Register

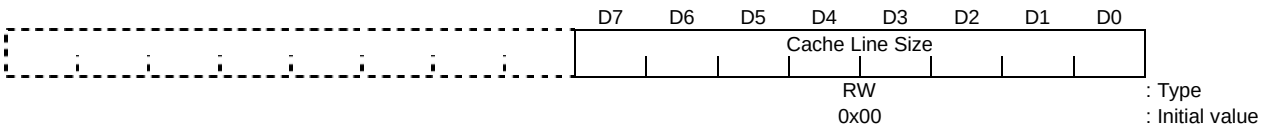
8.2.2.6 Class Code Register (0x09–0x0B)



Bits	Mnemonic	Field Name	Description
D23:D16	BASECLASS CODE	Base Class Code	Class Code This value 0x0C03FE represents the USB Device Controller.
D15:D8	SUBCLASSC ODE	Sub Class Code	
D7:D0	INTERFACE	Interface	

Figure 8.2.7 Class Code Register

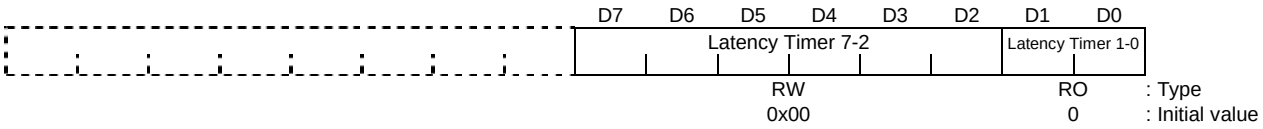
8.2.2.7 Cache Line Size Register (0x0C)



Bits	Mnemonic	Field Name	Description
D7:D0	Cache Line Size	Cache Line Size	Cache Line Size This register specifies the system cache line size in units of DWORDs.

Figure 8.2.8 Cache Line Size Register

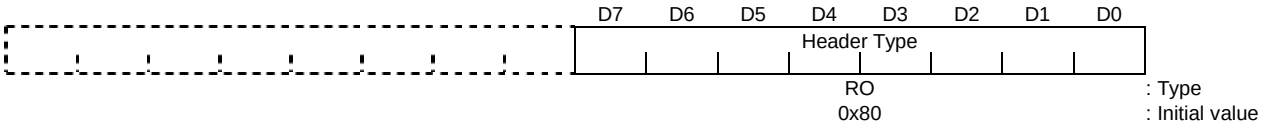
8.2.2.8 Latency Timer Register (0x0D)



Bits	Mnemonic	Field Name	Description
D7:D0	Latency Timer	Latency Timer	Latency Timer This register specifies, in units of PCI bus clocks, the value of the Latency Timer for this PCI bus master.

Figure 8.2.9 Latency Timer Register

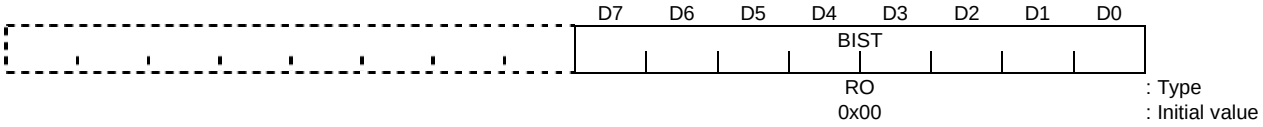
8.2.2.9 Header Type Register (0x0E)



Bits	Mnemonic	Field Name	Description
D7:D0	Header Type	Header Type	Header Type This value 0x80 represents a Multifunction device.

Figure 8.2.10 Header Type Register

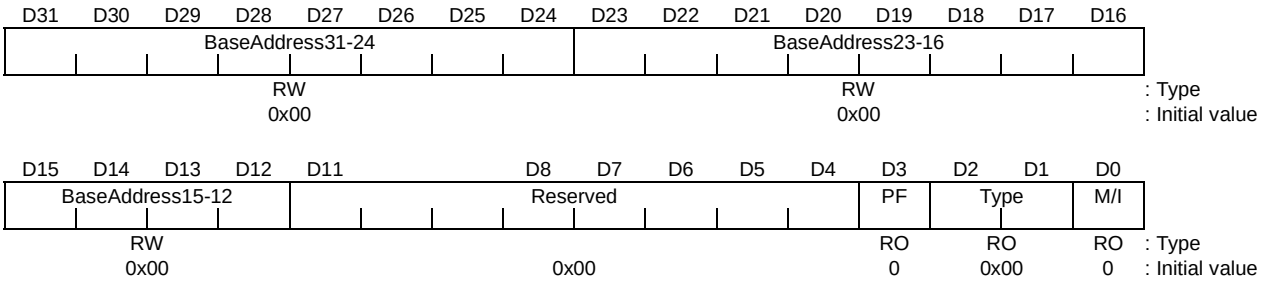
8.2.2.10 Built-in Self Test Register (0x0F)



Bits	Mnemonic	Field Name	Description
D7:D0	BIST	Built-in Self Test	Built-in Self Test Fixed to “0” since BIST is not supported.

Figure 8.2.11 Built-in Self Test Register

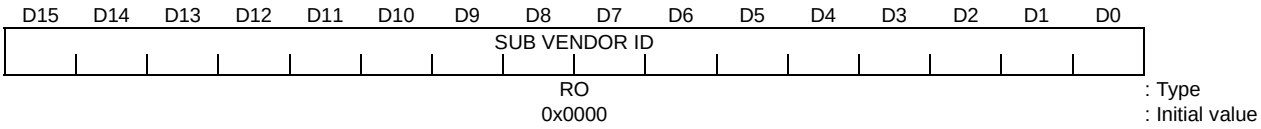
8.2.2.11 UDC Memory Base Address (0x10-0x13)



Bits	Mnemonic	Field Name	Description
D31:D12	BaseAddress	Base Address	Base Address These bits comprise the upper 20 bits of the base address used for accessing UDC registers.
D11:D4		Reserved	Reserved Fixed to “0” since UDC registers need to be mapped on 4 KB boundaries.
D3	PF	Prefetchable	Prefetchable Fixed to “0” since UDC registers must be allocated to a non-cacheable area.
D2:D1	Type	Type	Type Fixed to “00” since UDC registers are 32-bit address space.
D0	M/I	Memory space Indicator	Memory space Indicator Fixed to “0” since the base address register is used for memory.

Figure 8.2.12 UDC Memory Base Address

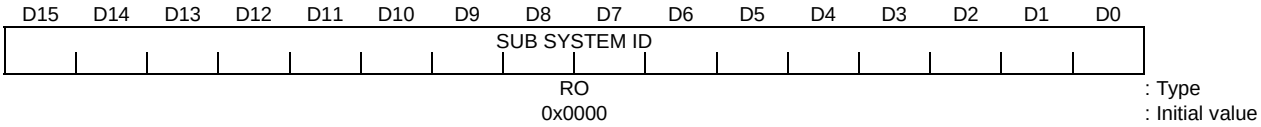
8.2.2.12 Sub Vendor ID Register (0x2C–0x2D)



Bits	Mnemonic	Field Name	Description
D15:D0	SUB VENDOR ID	Sub Vendor ID	Sub Vendor ID If use Power On Loading function, please refer to 10. Loadable PCI Configuration Space.

Figure 8.2.13 Sub Vendor ID Register

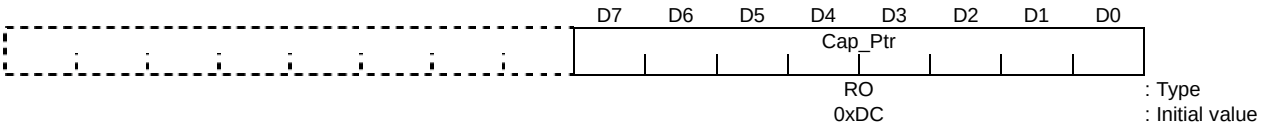
8.2.2.13 Sub System ID Register (0x2E–0x2F)



Bits	Mnemonic	Field Name	Description
D15:D0	SUB SYSTEM ID	Sub System ID	Sub System ID If use Power On Loading function, please refer to 10. Loadable PCI Configuration Space.

Figure 8.2.14 Sub System ID Register

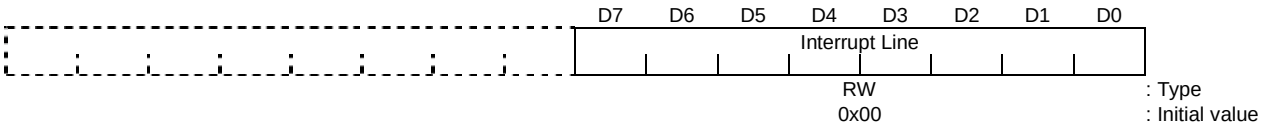
8.2.2.14 Capability Pointer Register (0x34)



Bits	Mnemonic	Field Name	Description
D7:D0	Cap_Ptr	Capability Pointer	Capability Pointer

Figure 8.2.15 Capability Pointer Register

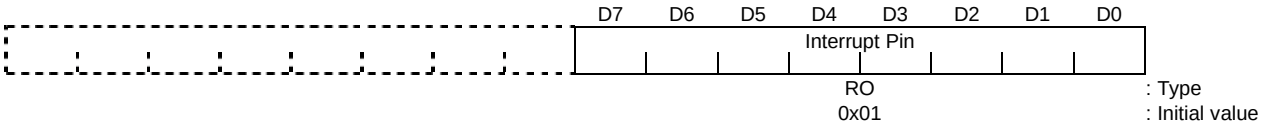
8.2.2.15 PCI Interrupt Line Register (0x3C)



Bits	Mnemonic	Field Name	Description
D7:D0	Interrupt Line	Interrupt Line	Interrupt Line These bits specify an IRQ interrupt number.

Figure 8.2.16 PCI Interrupt Line Register

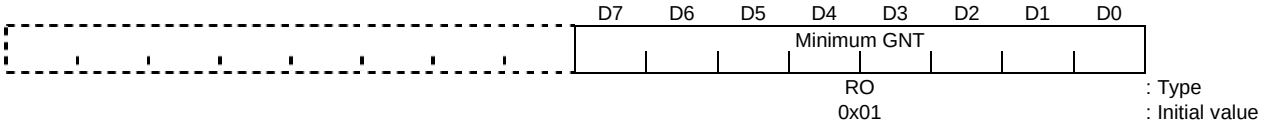
8.2.2.16 PCI Interrupt Pin Register (0x3D)



Bits	Mnemonic	Field Name	Description
D7:D0	Interrupt Pin	Interrupt Pin	Interrupt Pin This register tells which Interrupt PIN the device uses. It can be specified by the PCI Interrupt PIN Selectable Register (0x44).

Figure 8.2.17 PCI Interrupt Pin Register

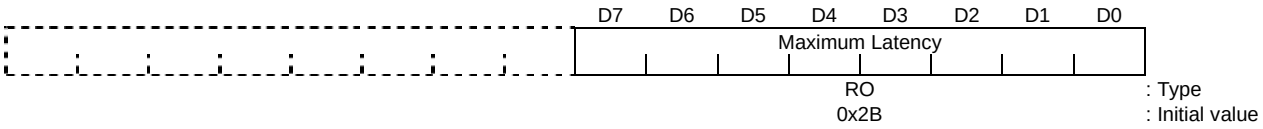
8.2.2.17 Minimum GNT Register (0x3E)



Bits	Mnemonic	Field Name	Description
D7:D0	Minimum GNT	Minimum GNT	Minimum GNT This is the minimum burst time that this module requires. Specify it in units of 0.25 µs. The minimum burst time for this module is fixed to “0x01.”

Figure 8.2.18 Minimum GNT Register

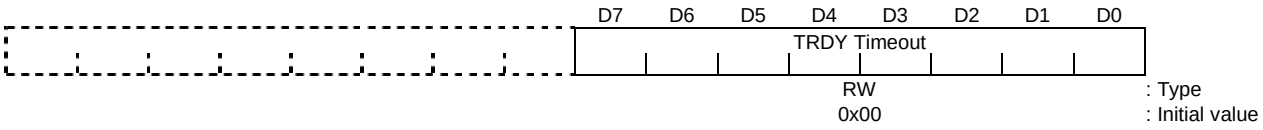
8.2.2.18 Maximum Latency Register (0x3F)



Bits	Mnemonic	Field Name	Description
D7:D0	Maximum Latency	Maximum Latency	Maximum Latency This is the maximum wait time during PCI access. Specify it in units of 0.25 µs. The maximum wait time for this module is fixed to 0x2B.

Figure 8.2.19 Maximum Latency Register

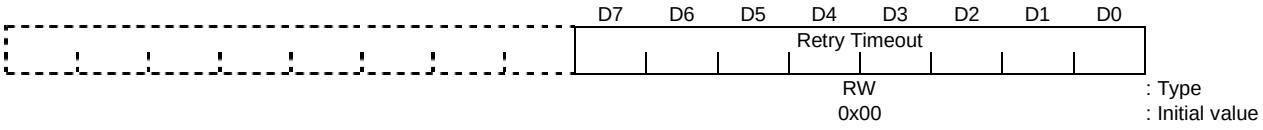
8.2.2.19 TRDY Timeout Register (0x40)



Bits	Mnemonic	Field Name	Description
D7:D0	TRDY Time out	TRDY Time out	TRDY Time out Value Sets number of PCI clocks that core as Master will wait for TRDY.

Figure 8.2.20 TRDY Timeout Register

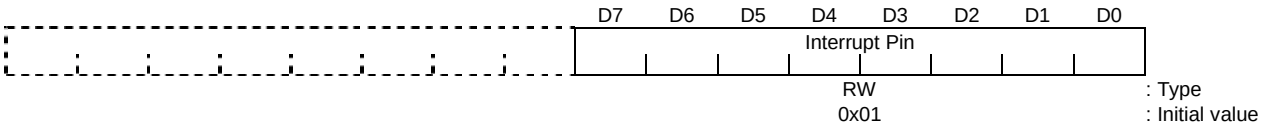
8.2.2.20 Retry Timeout Register (0x41)



Bits	Mnemonic	Field Name	Description
D7:D0	Retry Time out	Retry Time out	Retry Timeout Value Sets number of retries that the core as Master will perform.

Figure 8.2.21 Retry Timeout Register

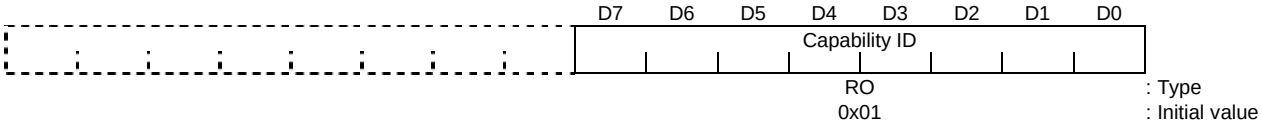
8.2.2.21 PCI Interrupt Pin Selectable Register (0x44)



Bits	Mnemonic	Field Name	Description
D7:D0	Interrupt Pin	Interrupt Pin	Interrupt Pin These bits set a PCI interrupt. 0x01: INTA* 0x02: INTB* Others: Default (INTA*)

Figure 8.2.22 PCI Interrupt Pin Selectable Register

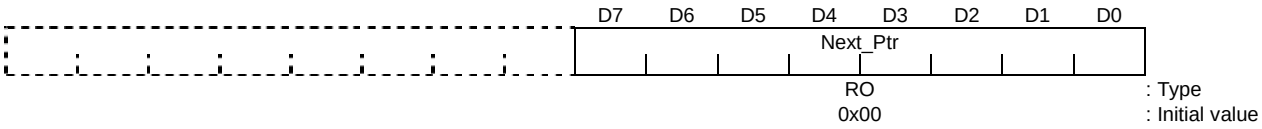
8.2.2.22 Capability ID Register (0xDC)



Bits	Mnemonic	Field Name	Description
D7:D0	Capability ID	Capability ID	Capability ID Fixed to “1.” Represents the PCI power management register block.

Figure 8.2.23 Capability ID Register

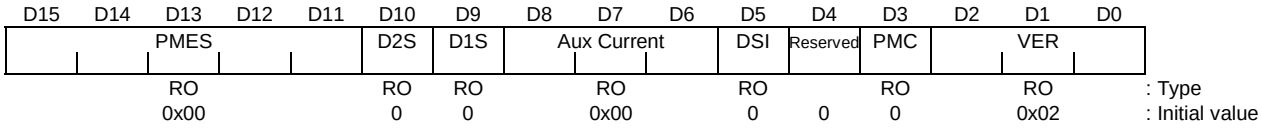
8.2.2.23 Next Item Pointer Register (0xDD)



Bits	Mnemonic	Field Name	Description
D7:D0	Next_Ptr	Next Item Pointer	Next Item Pointer Fixed to “0x00” since no next block exists.

Figure 8.2.24 Next Item Pointer Register

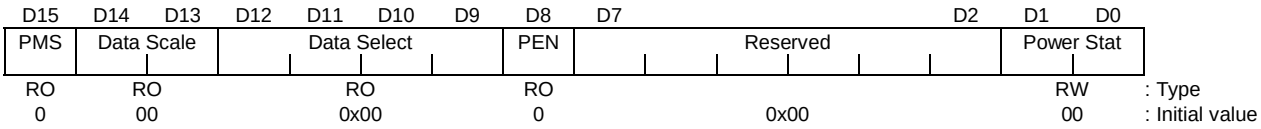
8.2.2.24 Power Management Capabilities PMC (0xDE–DF)



Bits	Mnemonic	Field Name	Description
D15:D11	PMES	PME Support	PME Support Set these bits to “0” since PME* is not supported.
D10	D2S	D2 Support	D2 Support Set this bit to “0” since D2 is not supported.
D9	D1S	D1 Support	D1 Support Set this bit to “0” since D1 is not supported.
D8:D6	Aux Current	Aux Current	Aux Current These bits set the auxiliary current value required for the 3.3 V auxiliary power supply. Fixed to “0” since PME* is not supported.
D5	DSI	Device Specific Initialization	Device Specific Initialization
D4		Reserved	
D3	PMC	PME Clock	PME Clock Clear this bit to “0” since PCI clock is not required to assert PME*.
D2:D0	VER	Version	Version These bits set a version number, which is “010b” indicating that the module conforms to PCI Power Management Interface Specifications 1.1.

Figure 8.2.25 Power Management Capabilities PMC

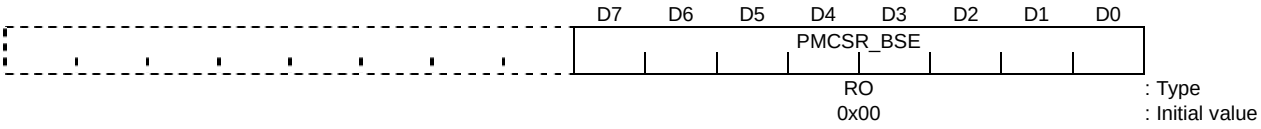
8.2.2.25 Power Management Control/Status Register PMCSR (0xE0–0xE1)



Bits	Mnemonic	Field Name	Description
D15	PMS	PME Status	PME Status Fixed to “0” since PME* is not supported.
D14:D13	Data Scale	Data Scale	Data Scale Fixed to “0x00” since Data Scale is not supported.
D12:D9	Data Select	Data Select	Data Select Fixed to “0x00” since Data Select is not supported.
D8	PEN	PME Enable	PME Enable Fixed to “0x00” since PME* is not supported.
D7:D2		Reserved	
D1:D0	Power Stat	Power Status	Power Status The OS uses these bits to set the current device status. 00: D0 11: D3

Figure 8.2.26 Power Management Control/Status Register PMCSR

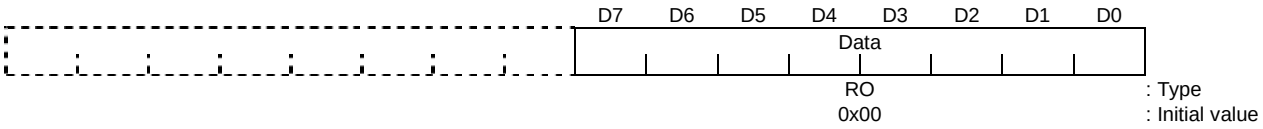
8.2.2.26 PCI to PCI Bridge Support Extension Register PMCSR BER (0xE2)



Bits	Mnemonic	Field Name	Description
D7:D0	PMCSR_BSE	MCSR_BSE	MCSR_BSE Set this bit to “0” since this module is not a PCI-PCI bridge.

Figure 8.2.27 PCI to PCI Bridge Support Extension Register PMCSR BER

8.2.2.27 Data (0xE3)



Bits	Mnemonic	Field Name	Description
D7:D0	Data	Data	Data Fixed to “0” since the Data Select is not supported.

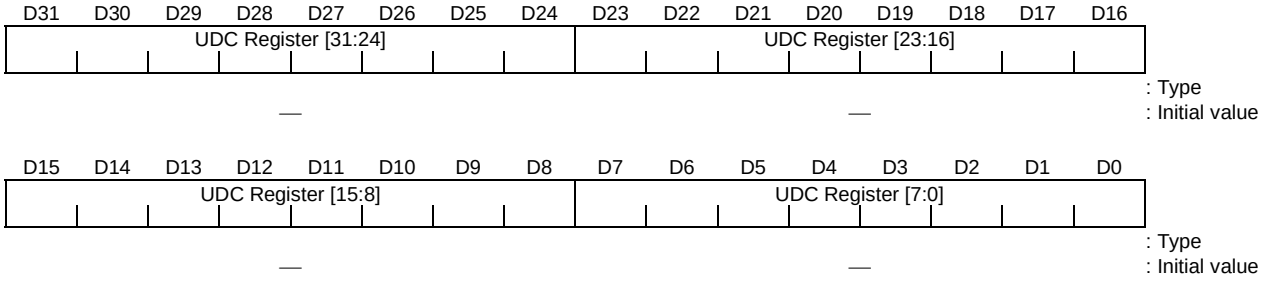
Figure 8.2.28 Data

8.2.3 UDC Registers

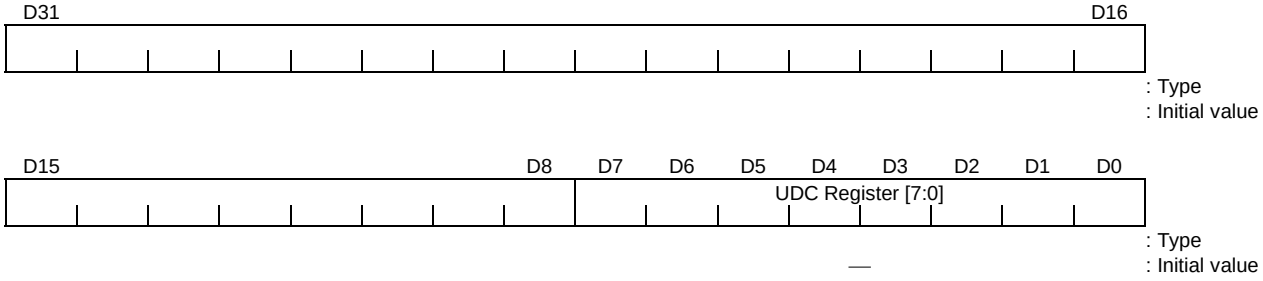
This module has in its memory space UDC registers listed in the following table. These registers can be accessed by specifying the base address in the Memory Base Address Register.

8.2.3.1 USB UDC Register Map

UDC Register (Offset 0x000-0x1FF) 32 bits register



UDC Register (Offset 0x200-0xFFF) 8 bits register is extended to 32 bits register.



Offset	Size (Bytes)	Name
0x00	4	INT_status register
0x04	4	INT_enable register
0x08	4	MST_setting register
0x0C	4	MSTwr_start_address register
0x10	4	MSTwr_end_address register
0x14	4	MSTwr_crt_address register
0x18	4	MSTrd_start_address register
0x1C	4	MSTrd_end_address register
0x20	4	MSTrd_crt_address register
0x24	4	Power_detect_control register
0x28-0x1FF	0x1D8	Reserved
0x200	4	ENDPOINT0 Tx, Rx FIFO
0x204	4	ENDPOINT1 Tx, Rx FIFO
0x208	4	ENDPOINT2 Tx, Rx FIFO
0x20C	4	ENDPOINT3 Tx, Rx FIFO
0x210-0x223	0x14	Reserved
0x224	4	EP1_MODE register
0x228	4	EP2_MODE register
0x22C	4	EP3_MODE register
0x230-0x23F	0x10	Reserved
0x240	4	EP0_STATUS register
0x244	4	EP1_STATUS register
0x248	4	EP2_STATUS register
0x24C	4	EP3_STATUS register
0x250-0x25F	0x10	Reserved
0x260	4	EP0_SIZE_L_A register
0x264	4	EP1_SIZE_L_A register
0x268	4	EP2_SIZE_L_A register
0x26C	4	EP3_SIZE_L_A register
0x270-0x283	0x14	Reserved
0x284	4	EP1_SIZE_L_B register
0x288	4	EP2_SIZE_L_B register
0x28C-0x2A3	0x18	Reserved
0x2A4	4	EP1_SIZE_H_A register
0x2A8	4	EP2_SIZE_H_A register
0x2AC	4	EP3_SIZE_H_A register
0x2B0-0x2C3	0x14	Reserved
0x2C4	4	EP1_SIZE_H_B register
0x2C8	4	EP2_SIZE_H_B register
0x2CC-0x2FF	0x34	Reserved
0x300	4	BmRequest Type register
0x304	4	BRequest register
0x308	4	wValue_L register
0x30C	4	wValue_H register
0x310	4	wIndex_L register
0x314	4	wIndex_H register
0x318	4	wLength_L register
0x31C	4	wLength_H register
0x320	4	SetupReceived register
0x324	4	Current_Config register
0x328	4	StandardRequest register
0x32C	4	Request register

Offset	Size (Bytes)	Name
0x330	4	Dataset register
0x334	4	Reserved
0x338	4	USB_STATE register
0x33C	4	EOP register
0x340	4	Command register
0x344	4	EPx_SINGLE register
0x348	4	Reserved
0x34C	4	EPx_BCS register
0x350-0x357	8	Reserved
0x358	4	INT_Control register
0x35C	4	Reserved
0x360	4	StandardRequestMode register
0x364	4	RequestMode register
0x368-0x37F	0x18	Reserved
0x380	4	Port_Status register
0x384-0x38B	8	Reserved
0x38C	4	ADDRESS register
0x390	4	Reserved
0x394	4	Reserved
0x398	4	USBREADY register
0x39C	4	Reserved
0x3A0	4	SetDescriptor STALL register
0x3A4-0x7FF	0x45C	Reserved
0x800	4	Descriptor_RAM offset 0x00
↓	↓	↓
0x9FC	4	Descriptor_RAM offset 0x7F
0xA00-0xFFF	0x600	Reserved

8.2.3.2 Interrupt Cause Status Register (0x00-0x03)

D31	...	D20	D19	D18	D17	D16	
Reserved			INT_PW detect	INT_ syserr	MSTrd_ end_add	MSTwr_ timeout	
RO			RW	RW	RW	RW	: Type
0x000			0	0	0	0	: Initial value

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	
MSTwr_ end_add	MSTwr_ set_add	ERR	SOF	INT_EP 3NAK	INT_EP 2NAK	INT_EP 1NAK	EP3_DA TASET	EP2_DA TASET	EP1_DA TASET	INT_STA TUSNAK	INT_ST ATUS	INT_SE TUP	INT_END POINT0	USB Reset	Suspend Resume	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	: Type
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	: Initial value

This register asserts "1" to each corresponding bit when an interrupt cause is generated. Writing "0" to each bit also makes it possible to clear the status.

Bits	Mnemonic	Field Name	Description
D31:D20		Reserved	
D19	INT_PW detect	INT_Power Detect – Power Supply Detection Interrupt Status	INT_Power Detect – Power Supply Detection Interrupt Status "1" is asserted when the status of the USBDPDECT pin changes, or when "1" is detected from USBDPDECT pin after hardware reset. Writing "0" also makes it possible to clear the status. 0: No change 1: Status changed
D18	INT_syserr	INT_system_error – System Error status	INT_system_error – System Error status "1" is asserted when an internal Bus error is detected. When this occurs, assert either a system reset or a power reset, and then initialize the status. In addition, writing "0" makes it possible to clear the status. 0: Undetected 1: System error
D17	MSTrd_end_ add	MSTrd_end_add – Master Read Transfer End Status	MSTrd_end_add – Master Read Transfer End Status If a timeout occurs during the Master Write transfer operation, "1" is asserted to this status. Also, writing "0" makes it possible to clear the status. 0: Undetected 1: Master Write transfer timeout
D16	MSTwr_ timeout	MSTwr_Timeout – Master Write Transfer Timeout Status	MSTwr_Timeout – Master Write Transfer Timeout Status If a timeout occurs during the Master Write transfer operation, "1" is asserted to this status. Also, writing "0" makes it possible to clear the status. 0: Undetected 1: Master Write transfer timeout
D15	MSTwr_end_ add	MSTwr_end_add – Master Write Transfer Timeout Status	MSTwr_end_add – Master Write Transfer Timeout Status "1" is asserted when Master Write transfer ends. Writing "0" also makes it possible to clear the status. 0: Undetected 1: Master Write Transfer end

Figure 8.2.29 Interrupt Cause Status Register (1/3)

Bits	Mnemonic	Field Name	Description
D14	MSTwr_set_add	MSTwr_set_add – Master Write Transfer Address Request Status	MSTwr_set_add – Master Write Transfer Address Request Status "1" is asserted when the Master Write Block requests a transfer address. Also, writing "0" makes it possible to clear the status. 0: Undetected 1: Master Write Transfer Address request
D13	ERR	ERR – UDC Protocol Error Interrupt Status	ERR – UDC Protocol Error Interrupt Status The UDC asserts "1" when protocol errors such as bit stuffing, CRC, or PID occur. Also, writing "0" makes it possible to clear the status. 0: Undetected 1: Error
D12	SOF	SOF – UDC SOF Interrupt Status	SOF – UDC SOF Interrupt Status "1" is asserted when SOF (Start of Frame) is detected. Writing "0" makes it possible toclear the status. 0: No SOF detected 1: SOF detected
D11	INT_EP3NAK	INT_EP3NAK – EP3NAK Interrupt Status	INT_EP3NAK – EP3NAK Interrupt Status "1" is asserted when INT_EP3NAK is detected. Also, writing "0" makes it possible to clear the status. 0: Undetected 1: Detect INT_EP3NAK
D10	INT_EP2NAK	INT_EP2NAK – EP2NAK Interrupt Status	INT_EP2NAK – EP2NAK Interrupt Status "1" is asserted when INT_EP2NAK is detected. Also, writing "0" makes it possible to clear the status. 0: Undetected 1: Detect INT_EP2NAK
D9	INT_EP1NAK	INT_EP1NAK – EP1NAK Interrupt Status	INT_EP1NAK – EP1NAK Interrupt Status "1" is asserted when INT_EP1NAK is detected. Also, writing "0" makes it possible to clear the status. 0: Undetected 1: Detect INT_EP1NAK
D8	EP3_ DATASET	EP3_DATASET – EP3 Data Set Interrupt Status	EP3_DATASET – EP3 Data Set Interrupt Status Detects the DATASET signal of endpoint 3. "1" is asserted at the falling edge of the DATASET signal when endpoint 3 is IN, or at the rising edge when it is OUT. Also, writing "0" makes it possible to clear the status. 0: Undetected 1: Detect EP3_DATASET
D7	EP2_ DATASET	EP2_DATASET – EP2 Data Set Interrupt Status	EP2_DATASET – EP2 Data Set Interrupt Status Detects the DATASET signal of endpoint 2. "1" is asserted at the falling edge of the DATASET signal when endpoint 2 is IN, or at the rising edge when it is OUT. Also, writing "0" makes it possible to clear the status. 0: Undetected 1: Detect EP2_DATASET
D6	EP1_ DATASET	EP1_DATASET – EP1 Data Set Interrupt Status	EP1_DATASET – EP1 Data Set Interrupt Status Detects the DATASET signal of endpoint 1. "1" is asserted at the falling edge of the DATASET signal when endpoint 1 is IN, or at the rising edge when it is OUT. Also, writing "0" makes it possible to clear the status. 0: Undetected 1: Detect EP1_DATASET

Figure 8.2.29 Interrupt Cause Status Register (2/3)

Bits	Mnemonic	Field Name	Description
D5	INT_ STATUSNAK	INT_ STATUSNAK – INT_ STATUSNAK Interrupt Status	INT_STATUSNAK – INT_STATUSNAK Interrupt Status "1" is asserted when INT_STATUSNAK is detected. Also, writing "0" makes it possible to clear the status. When using this interrupt, please enable the UDC Interrupt Control Register as well. 0: Undetected 1: Detect INT_STATUSNAK
D4	INT_STATUS	INT_STATUS – INT_STATUS Interrupt Status	INT_STATUS – INT_STATUS Interrupt Status "1" is asserted when INT_STATUS is detected. Also, writing "0" makes it possible to clear the status. 0: Undetected 1: Detect INT_STATUSNAK
D3	INT_SETUP	INT_SETUP – INT_SETUP Interrupt Status	INT_SETUP – INT_SETUP Interrupt Status "1" is asserted when INT_SETUP is detected. Also, writing "0" makes it possible to clear the status. The SetupReceived Register of the UDC is cleared when the status is cleared. 0: Undetected 1: Detect INT_SETUP
D2	INT_ ENDPOINT0	INT_ ENDPOINT0 – INT_ENDPOINT0 Interrupt Status	INT_ENDPOINT0 – INT_ENDPOINT0 Interrupt Status "1" is asserted when INT_ENDPOINT0 is detected. Also, writing "0" makes it possible to clear the status. 0: Undetected 1: Detect INT_ENDPOINT0
D1	USB Reset	USB_RESET – USB_RESET Interrupt Status	USB_RESET – USB_RESET Interrupt Status "1" is asserted when USB_RESET is detected. Also, writing "0" makes it possible to clear the status. 0: Undetected 1: Detect USB_RESET
D0	Suspend Resume	Suspend Resume – Suspend/Resume Interrupt Status	Suspend Resume – Suspend/Resume Interrupt Status "1" is asserted each time the UDC SUSPEND signal changes. Also, writing "0" makes it possible to clear the status. 0: Undetected 1: Status changed

Figure 8.2.29 Interrupt Cause Status Register (3/3)

8.2.3.3 Interrupt Cause Enable Register (0x04-0x07)

D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16
Reserved												INT_PW detect	INT_ syserr	MSTrd_ end_add	MSTwr_ timeout
RO												RW	RW	RW	RW : Type
0x000												0	0	0	0 : Initial value

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
MSTwr_ end_add	MSTwr_ set_add	ERR	SOF	INT_EP 3NAK	INT_EP 2NAK	INT_EP 1NAK	EP3_DA TASET	EP2_DA TASET	EP1_DA TASET	INT_ STATUS NAK	INT_ STATUS	INT_ SETUP	INT_END POINT0	USB Reset	Suspend Resume
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RO	RW	RW	RW	RW	RW : Type
0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0 : Initial value

Bits	Mnemonic	Field Name	Description
D31:D20		Reserved	
D19	INT_PW detect	INT_Power Detect – Power Detect Interrupt Cause Enable	INT_Power Detect – Power Detect Interrupt Cause Enable Controls INT_PowerDetect interrupts. 0: Disable 1: Enable
D18	INT_syserr	INT_system_error – System Error Status Cause Enable	INT_system_error – System Error Status Cause Enable Controls INT_system_error interrupts. 0: Disable 1: Enable
D17	MSTrd_end_ add	MSTrd_end_add – Master Read Transfer End Status Cause Enable	MSTrd_end_add – Master Read Transfer End Status Cause Enable Controls MSTrd_end_add interrupts. 0: Disable 1: Enable
D16	MSTwr_t imeout	MSTwr_Timeout – Master Write Transfer Time Out Status Cause Enable	MSTwr_Timeout – Master Write Transfer Time Out Status Cause Enable Controls MSTwr_Timeout interrupts. 0: Disable 1: Enable
D15	MSTwr_end_ add	MSTwr_end_add – Master Write Transfer End Status Cause Enable	MSTwr_end_add – Master Write Transfer End Status Cause Enable Controls MSTwr_end_add interrupts. 0: Disable 1: Enable
D14	MSTwr_set_ add	MSTwr_set_add – Master Write Transfer Address Request Status Cause Enable	MSTwr_set_add – Master Write Transfer Address Request Status Cause Enable Controls MSTwr_set_add interrupts. 0: Disable 1: Enable
D13	ERR	ERR – UDC Error Interrupt Cause Enable	ERR – UDC Error Interrupt Cause Enable Controls ERR interrupts. 0: Disable 1: Enable

Figure 8.2.30 Interrupt Cause Enable Register (1/3)

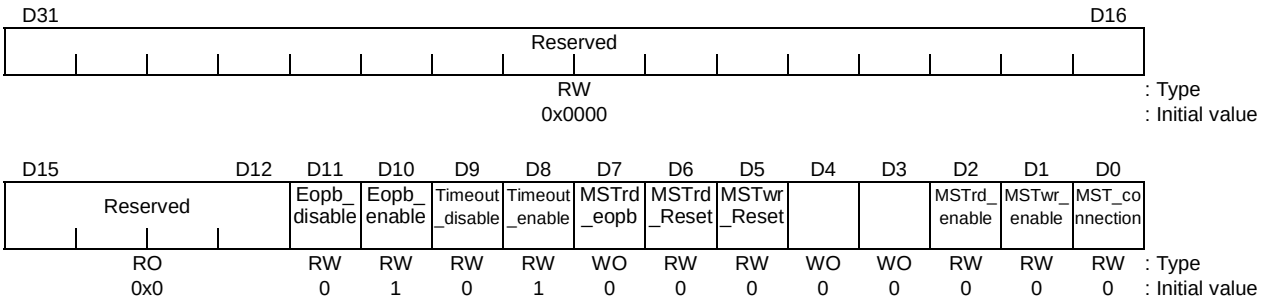
Bits	Mnemonic	Field Name	Description
D12	SOF	SOF – UDC SOF Interrupt Cause Enable	SOF – UDC SOF Interrupt Cause Enable Controls SOF interrupts. 0: Disable 1: Enable
D11	INT_EP3NAK	INT_EP3NAK – EP3NAK Interrupt Cause Enable	INT_EP3NAK – EP3NAK Interrupt Cause Enable Controls INT_EP3NAK interrupts. 0: Disable 1: Enable
D10	INT_EP2NAK	INT_EP2NAK – EP2NAK Interrupt Cause Enable	INT_EP2NAK – EP2NAK Interrupt Cause Enable Controls INT_EP2NAK interrupts. 0: Disable 1: Enable
D9	INT_EP1NAK	INT_EP1NAK – EP1NAK Interrupt Cause Enable	INT_EP1NAK – EP1NAK Interrupt Cause Enable Controls INT_EP1NAK interrupts. 0: Disable 1: Enable
D8	EP3_ DATASET	EP3_DATASET – EP3 Data Set Interrupt Cause Enable	EP3_DATASET – EP3 Data Set Interrupt Cause Enable Controls EP3_DATASET interrupts. 0: Disable 1: Enable
D7	EP2_ DATASET	EP2_DATASET – EP2 Data Set Interrupt Cause Enable	EP2_DATASET – EP2 Data Set Interrupt Cause Enable Controls EP2_DATASET interrupts. 0: Disable 1: Enable
D6	EP1_ DATASET	EP1_DATASET – EP1 Data Set Interrupt Cause Enable	EP1_DATASET – EP1 Data Set Interrupt Cause Enable Controls EP1_DATASET interrupts. 0: Disable 1: Enable
D5	INT_ STATUSNAK	INT_ STATUSNAK – INT_ STATUSNAK Interrupt Cause Enable	INT_STATUSNAK – INT_STATUSNAK Interrupt Cause Enable This read-only interrupt is always enabled. Please use the Interrupt Control Register in the UDC unit to control the INT_STATUSNAK interrupt. 1: Enable (always set)
D4	INT_STATUS	INT_STATUS – INT_STATUS Interrupt Cause Enable	INT_STATUS – INT_STATUS Interrupt Cause Enable Controls INT_STATUS interrupts. 0: Disable 1: Enable
D3	INT_SETUP	INT_SETUP – INT_SETUP Interrupt Cause Enable	INT_SETUP – INT_SETUP Interrupt Cause Enable Controls INT_SETUP interrupts. 0: Disable 1: Enable
D2	INT_ ENDPOINT0	INT_ ENDPOINT0 – INT_ENDPOINT0 Interrupt Cause Enable	INT_ENDPOINT0 – INT_ENDPOINT0 Interrupt Cause Enable Controls INT_ENDPOINT0 interrupts. 0: Disable 1: Enable

Figure 8.2.30 Interrupt Cause Enable Register (2/3)

Bits	Mnemonic	Field Name	Description
D1	USB Reset	USB_RESET – USB_RESET Interrupt Cause Enable	USB_RESET – USB_RESET Interrupt Cause Enable Controls USB_RESET interrupts. 0: Disable 1: Enable
D0	Suspend Resume	Suspend Resume – Suspend/Resume Interrupt Cause Enable	Suspend Resume – Suspend/Resume Interrupt Cause Enable Controls SuspendResume interrupts. 0: Disable 1: Enable

Figure 8.2.30 Interrupt Cause Enable Register (3/3)

8.2.3.4 Master Operation Setting Register (0x08-0x0B)



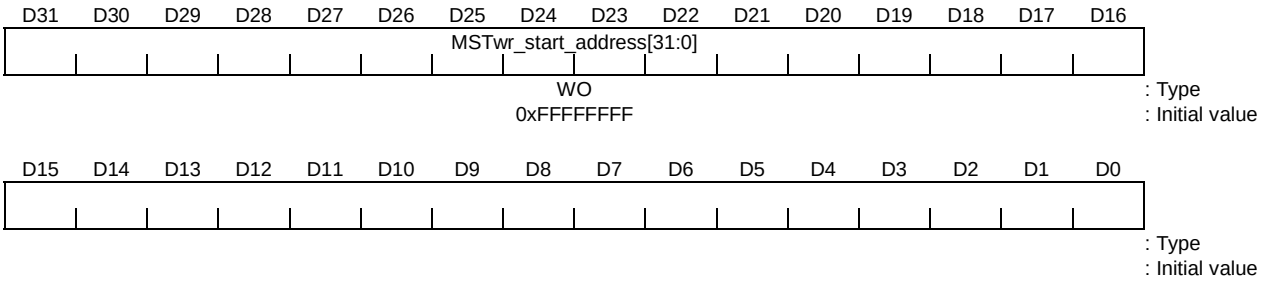
Bits	Mnemonic	Field Name	Description
D31:D16		Reserved	Please set these bits to 0x0000.
D15:D12		Reserved	
D11	eopb_ disable	eopb_disable – Master Read EOP disable	eopb_disable - Master read EOP disable Master read EOP is enabled in initial state. Don't change this bit setting in master read transfer. It will be set to "Enable" if both Enable and Disable are selected. 0: No operation 1: Disable Master read EOP
D10	eopb_ enable	eopb_enable – Master Read EOP enable	eopb_enable- Master read EOP enable Master read EOP is enabled in initial state. Don't change this bit setting in master read transfer. It will be set to "Enable" if both Enable and Disable are selected. 0: No operation 1: Enable Master read EOP
D9	Timeout_ disable	Timeout_disable – Timeout disable	Timeout_disable- Master timeout disable Master timeout is enabled in initial state. Don't change this bit setting in master write transfer. It will be set to "Enable" if both Enable and Disable are selected. 0: No operation 1: Disable Time out
D8	Timeout_ enable	Timeout_enable – Timeout enable	Timeout_enable - Master timeout enable Master timeout is enabled in initial state. Don't change this bit setting in master write transfer. It will be set to "Enable" if both Enable and Disable are selected. 0: No operation 1: Enable Time out
D7	MSTrd_eopb	MSTrd_eopb – Master Read EOP	MSTrd_eopb – Master Read EOP This bit is used for transmitting Null packets at endpoints connected to the Master Read operation side, and for transmitting short packets. When transmitting Null packets however, this bit is only valid when both DATASET_A and DATASET_B of the applicable endpoint are "0". This bit is ignored otherwise. In addition, this bit is only asserted for 1 clock at 12 MHz. 0: No operation 1: Assert EOP
D6	MSTrd_Reset	MSTrd_Reset – Master Read Reset	MSTrd_Reset – Master Read Reset Initializes the Master read (UDC becomes the data receiving side) operation block. However, it is necessary to initialize the corresponding endpoint from the UDC Command Register after this reset since FIFO (which is an endpoint inside the UDC) is not initialized. Please use this reset after stopping Master Read operation. 0: No operation 1: Reset

Figure 8.2.31 Master Operation Setting Register (1/2)

Bits	Mnemonic	Field Name	Description
D5	MSTwr_ Reset	MSTwr_Reset – Master Write Reset	MSTwr_Reset – Master Write Reset Initializes the Master Write (UDC becomes the data sending side) operation block. However, it is necessary to initialize the corresponding endpoint from the UDC Command Register before this reset since FIFO (which is an endpoint inside the UDC) is not initialized. Please use this reset .after stopping Master Write operation. 0: No operation 1: Reset
D4			Please always set this bit to “0”.
D3			Please always set this bit to “0”.
D2	MSTrd_ enable	MSTrd_enable – Master Read Enable	MSTrd_enable – Master Read Enable Controls the Master read (UDC becomes the data receiving side) operation. Please enable this register when the setting of the transfer address is complete. This bit is automatically disabled when Master read transfer ends. However can’t use this register to disable Master read operation. 0: Disable 1: Enable
D1	MSTwr_ enable	MSTwr_enable – Master Write Enable	MSTwr_enable – Master Write Enable Controls the Master write (UDC becomes the data sending side) operation. Please enable this register when the setting of the transfer address is complete. This bit is automatically disabled when Master write transfer ends. However can’t use this register to disable Master write operation. 0: Disable 1: Enable
D0	MST_ connection	MST_connection – Master Connection Setting	MST_connection – Master Connection Setting Sets the connection to the Master and the UDC endpoint. The Master write block is connected to the OUT operation endpoint, and the Master read block is connected to the IN operation endpoint. Only set this bit when setting up the UDC Registers in the USB DEVICE Controller setup state. Do not change this bit at other timing. 0: ENDPOINT1 (OUT), ENDPOINT2 (IN) 1: ENDPOINT1 (IN), ENDPOINT2 (OUT)

Figure 8.2.31 Master Operation Setting Register (2/2)

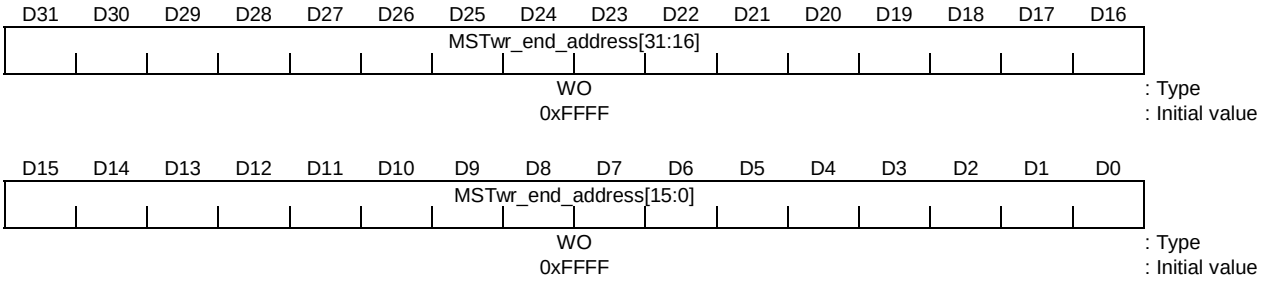
8.2.3.5 Master Write Start Address Setting Register (0x0C-0x0F)



Bits	Mnemonic	Field Name	Description
D31:D0	MSTwr_start_address[31:0]	MSTwr_start_address – Master Write Start Address	MSTwr_start_address – Master Write Start Address Please set the Master Write Start Address. However, this Master operation only supports increasing addresses, so please set a value lower than the Master Write End Address.

Figure 8.2.32 Master Write Start Address Setting Register

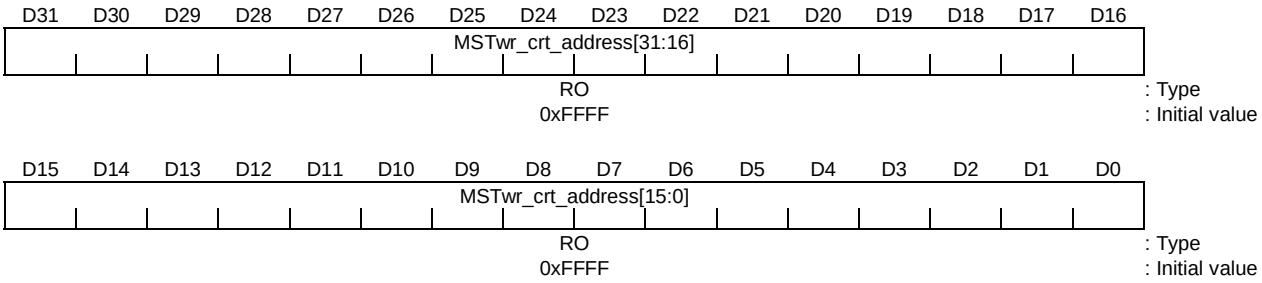
8.2.3.6 Master Write End Address Setting Register (0x10-0x13)



Bits	Mnemonic	Field Name	Description
D31:D0	MSTwr_end_address[31:0]	MSTwr_end_address – Master Write End Address	MSTwr_end_address – Master Write End Address Please set the Master Write End Address. However, this Master operation only supports increasing addresses, so please set a value higher than the Master Write Start Address.

Figure 8.2.33 Master Write End Address Setting Register

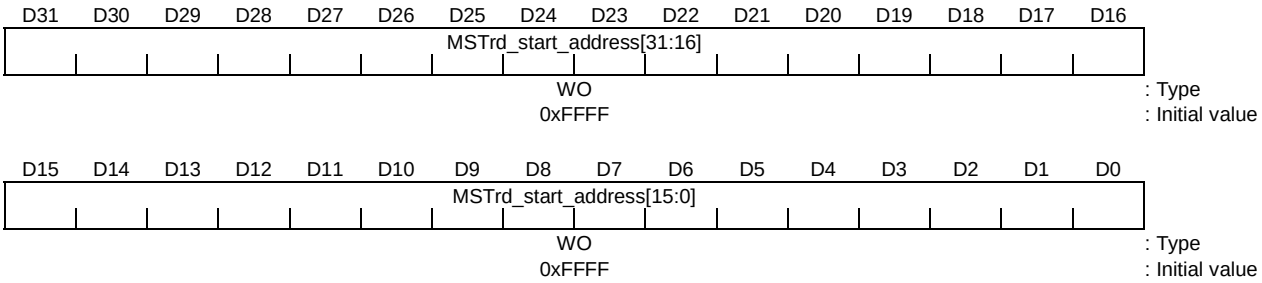
8.2.3.7 Master Write Current Address Register (0x14-0x17)



Bits	Mnemonic	Field Name	Description
D31:D0	MSTwr_crt_ address[31:0]	MSTwr_crt_ address – Master Write Current Address	MSTwr_crt_address – Master Write Current Address Displays the addresses of transfers performed during Master Write transfer to the target device that have completed. This field can be used when a Timeout interrupt occurs or when an error occurs during the transfer. This address is incremented when data is set from the endpoint to the Master Write 16-byte FIFO. While a Master Write transfer is in progress, data up to the displayed address will then exist in either the target device or in Master Write FIFO. In order to make the displayed address and the transfer completion address to the target device match, use either the Master Operation Setting Register or an interrupt to confirm that Master Write transfer is complete. Or, when confirming this while transfer is in progress, data transfer up to about 16 bytes may not be complete.

Figure 8.2.34 Master Write Current Address Register

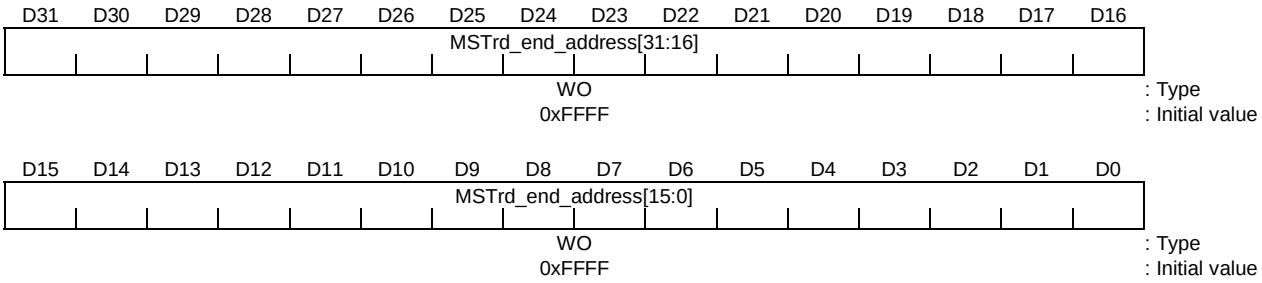
8.2.3.8 Master Read Start Address Setting Register (0x18-0x1B)



Bits	Mnemonic	Field Name	Description
D31:D0	MSTrd_start_ address[31:0]	MSTrd_start_ address – Master Read Start Address	MSTrd_start_address – Master Read Start Address Please set the Master Read Start Address. However, this Master operation only supports increasing addresses, so please set a value lower than the Master Read Start Address.

Figure 8.2.35 Master Read Start Address Setting Register

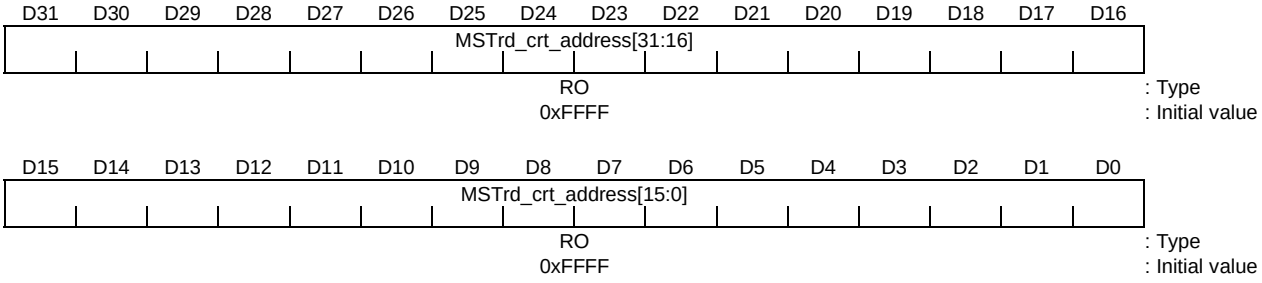
8.2.3.9 Master Read End Address Setting Register (0x1C-0x1F)



Bits	Mnemonic	Field Name	Description
D31:D0	MSTrd_end_ address[31:0]	MSTrd_end_ address – Master Read End Address	MSTrd_end_address – Master Read End Address Please set the Master Read End Address. However, this Master operation only supports increasing addresses, so please set a value higher than the Master Read Start Address.

Figure 8.2.36 Master Read End Address Setting Register

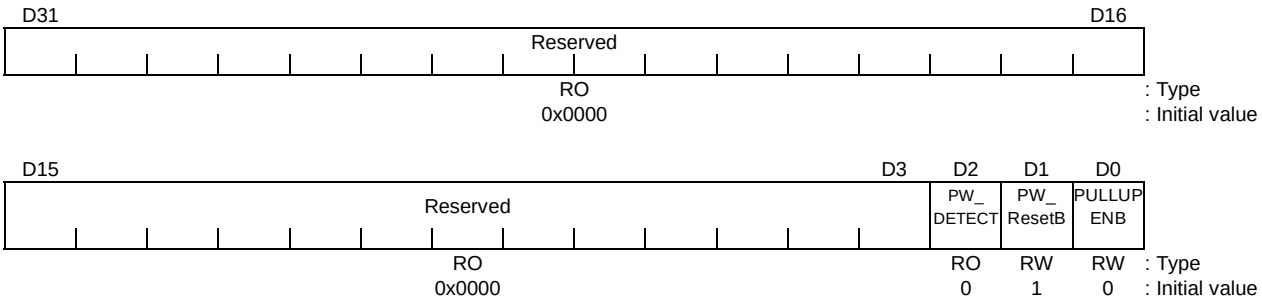
8.2.3.10 Master Read Current Address Register (0x20-0x23)



Bits	Mnemonic	Field Name	Description
D31:D0	MSTrd_crt_ address[31:0]	MSTrd_crt_ address – Master Read Current Address	MSTrd_crt_address – Master Read Current Address Please set the Master Read Start address. However, please set a physical address value that is lower than the Master Read End Address since this master operation only supports address increases. This address is incremented when data is set from Master Read 16-byte FIFO to the endpoint. While a Master Read transfer is in progress, data up to the displayed address will then exist in either the Master Write FIFO or in the endpoint FIFO. In order to make the displayed address and the transfer completion address from the target device match, use either the Master Operation Setting Register or an interrupt to confirm that Master Read transfer is complete. If a stall occurs during USB Host-Device communication, data transfer up to about 128 bytes during Bulk transfer may not be complete.

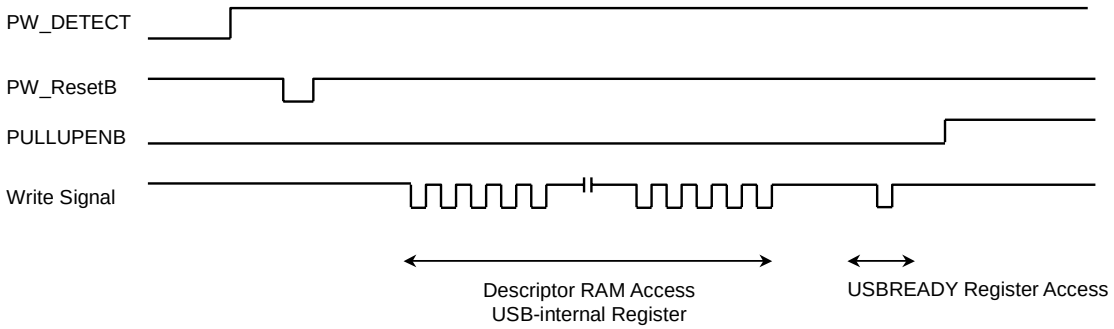
Figure 8.2.37 Master Read Current Address Register

8.2.3.11 Power Supply Detect Control Register (0x24-0x27)



Bits	Mnemonic	Field Name	Description
D31:D3		Reserved	
D2	PW_DETECT	PW_DETECT– USB Bus Power Supply Detect	PW_DETECT– USB Bus Power Supply Detect Detects the USB Bus power supply. (USBDPDECT pin) 0: Do not detect. 1: Detect USB Bus power supply
D1	PW_ResetB	PW_ResetB – Power Reset	PW_ResetB – Power Reset This software reset is used when detecting the USB Bus power supply. This reset initializes the Power Supply Detect Control Register, Interrupt Cause Enable Register, and all blocks other than the slave access circuit. Always be sure to deassert this reset since it is not cleared automatically. 0: Assert reset. 1: Deassert reset.
D0	PULLUPENB	PULLUPENB – USB Pull Up Setting	PULLUPENB – USB Pull Up Setting This setting is used for controlling USB data line pull up. This register value is output by the USBDPULLUP pin. 0: Hi-Z 1: PULLUP

Figure 8.2.38 Power Supply Detect Control Register



Please insert a sequence that detects the VDD signal level from the USB cable then initializes. At this time, the UDC disables USB_RESET signal detection from when PW_ResetB is cleared until when “0” is written to the USBREADY Register.

8.2.3.12 EPx_FIFO Register (0x200–0x20F)

EP0_FIFO Register (Endpoint0) (0x200)

D7	D6	D5	D4	D3	D2	D1	D0
EP0_DATA							
RW							: Type
—							: Initial value

EP1_FIFO Register (Endpoint1) (0x204)

D7	D6	D5	D4	D3	D2	D1	D0
EP1_DATA							
RW							: Type
—							: Initial value

EP2_FIFO Register (Endpoint2) (0x208)

D7	D6	D5	D4	D3	D2	D1	D0
EP2_DATA							
RW							: Type
—							: Initial value

EP3_FIFO Register (Endpoint3) (0x20C)

D7	D6	D5	D4	D3	D2	D1	D0
EP3_DATA							
RW							: Type
—							: Initial value

Figure 8.2.39 EPx_FIFO Register

This register is independently prepared for each endpoint. The CPU and DMA transfer data to and from the FIFO inside the UDC via this register.

Endpoint0 and Endpoint3 of this FIFO Register are only used for access from the CPU. For Enpoint1 and Endpoint2 however, access is also possible from local bus lines that can perform DMA transfer.

DMA transfer can be performed by setting the EPx_BCS Register to a value that can be used for DMA access. When accessing from the CPU bus, please set the EPx_BCS Register to a value that can be used for CPU access.

During enumeration, the REQUEST CONTROLLER inside the UDC automatically sets the mode that is defined in the Endpoint Descriptor for each endpoint. Each endpoint is automatically set to a particular direction from this set value.

8.2.3.13 EPx_MODE Register (0x224-0x22F)

This register sets the transfer mode of the endpoints.

When the software is specified to control the SET_CONFIG, SET_INTERFACE process, it is necessary to configure according to the specified Config or interface. Please access this register and set the mode when necessary. Please note that restrictions have been established regarding the timing at which Write operations are allowed.

When the software is specified to control processes for SET_CONFIG, SET_INTERFACE requests, complete the Write operation during the time between when INT_SETUP is received and the EOP Register is accessed. Any other timing will be a Write disabled timing and will be ignored.

EP1_MODE Register (Endpoint1) (0x204)

D7	D6	D5	D4	D3	D2	D1	D0	
Reserved		Payload			Mode		Direction	
			RW		RW		RW	: Type
00			0x00		00		0	: Initial value

EP2_MODE Register (Endpoint2) (0x208)

D7	D6	D5	D4	D3	D2	D1	D0	
Reserved		Payload			Mode		Direction	
			RW		RW		RW	: Type
00			0x00		00		0	: Initial value

EP3_MODE Register (Endpoint3) (0x22C)

D7	D6	D5	D4	D3	D2	D1	D0	
Reserved		Payload			Mode		Direction	
			RW		RW		RW	: Type
00			0x00		00		0	: Initial value

Bits	Mnemonic	Field Name	Description
D7:D6		Reserved	
D5:D3	Payload	PAYLOAD	PAYLOAD 0x00: 8 bytes (Endpoint 1,2,3) 0x01: 16 bytes (Endpoint 1,2,3) 0x02: 32 bytes (Endpoint 1,2,3) 0x03: 64 bytes (Endpoint 1,2,3) When a value other than 8, 16, 32 or 64 is specified as the wMaxPacketSize of the Descriptor, the automatic response of Set_Configuration, Set_Interface sets a Payload greater than the Descriptor value.
D2:D1	Mode	MODE	MODE 0x00: Control Transfer 0x01: Reserved 0x02: Bulk Transfer, or Interrupt Transfer 0x03: Interrupt (no toggling)
D0	Direction	DIRECTION	DIRECTION 0: OUT From the Host to the device 1: IN From the device to the Host

Figure 8.2.40 EPx_MODE Register

8.2.3.14 EPx_STATUS Register (0x240-0x24F)

EP0_STATUS Register (Endpoint0) (0x240)

D7	D6	D5	D4	D3	D2	D1	D0	
Reserved	TOGGLE	SUSPEND	STATUS			FIFO_ DISABLE	STAGE_ ERR	
RO	RO	RO	RO			RO	RO	: Type
0	0	0	111			0	0	: Initial value

EP1_STATUS Register (Endpoint1) (0x244)

D7	D6	D5	D4	D3	D2	D1	D0	
Reserved	TOGGLE	SUSPEND	STATUS			FIFO_ DISABLE	STAGE_ ERR	
RO	RO	RO	RO			RO	RO	: Type
0	0	0	111			0	0	: Initial value

EP2_STATUS Register (Endpoint2) (0x248)

D7	D6	D5	D4	D3	D2	D1	D0	
Reserved	TOGGLE	SUSPEND	STATUS			FIFO_ DISABLE	STAGE_ ERR	
RO	RO	RO	RO			RO	RO	: Type
0	0	0	111			0	0	: Initial value

EP3_STATUS Register (Endpoint3) (0x24C)

D7	D6	D5	D4	D3	D2	D1	D0	
Reserved	TOGGLE	SUSPEND	STATUS			FIFO_ DISABLE	STAGE_ ERR	
RO	RO	RO	RO			RO	RO	: Type
0	0	0	111			0	0	: Initial value

Bits	Mnemonic	Field Name	Description
D7		Reserved	
D6	TOGGLE	TOGGLE Bit	TOGGLE Bit Indicates the status of the Toggle Sequence bit. 0: TOGGLE BIT0 1: TOGGLE BIT1
D5	SUSPEND	SUSPEND	SUSPEND Indicates the UDC power management status. Access to the UDC is restricted when in the SUSPEND state. 0: RESUME 1: SUSPEND

Figure 8.2.41 EPx_STATUS Register (1/3)

Bits	Mnemonic	Field Name	Description
D4:D2	STATUS	ENDPOINT STATUS	ENDPOINT STATUS Indicates the status of the UDC endpoint. Displays whether or not transfer is performed for each endpoint, or displays the results of each transfer. This status is closely related to the transfer mode of each endpoint. The following table describes each state. 0x00: READY Receive: Device is ready to receive data. If SET_CONFIGURATION is used to define the endpoint transfer mode for Endpoint1 through Endpoint3, this register is initialized to the READY state. For Endpoint0, this register is initialized to the READY state when a USB reset from the Host is detected. This register is also set to READY when the Status Stage ends normally. Transmit: Is set when initialization similar to that for Receive above is performed. In the case of Transmit however, the Status Register will remain in the READY state even if data is set in the Transmit FIFO, data is normally transferred to the Host in response to a token from the Host, and then ACK is received. In this situation, a CPU interrupt is generated and the Data Set pin is deasserted. This makes it possible to signal that transmission ended normally at an external device. 0x01: DATAIN The UDC sets DATAIN in the Status Register along with an interrupt to the CPU when reception data from the Host is transferred to the FIFO without any error. In addition, the External Data Set pin is simultaneously asserted. 0x02: FULL 0x03: TX_ERR The UDC sets TX_ERR in the Status Register if ACK from the Host was not received after data was transmitted to an IN token from the Host. No interrupts to the CPU are generated in this situation. Only a status update is generated in this case since the Data Set pin is not deasserted either. The Host performs a retry then the IN token is sent again. 0x04: RX_ERR If there is an error (CRC error, etc.) in the data portion of the received token, the UDC sets RX_ERR in the Status Register without sending ACK to the Host. No interrupt is generated for the CPU in this case. Since the Data Set pin also is not asserted, only the Status is updated in this case. The Host performs a Retry operation then data is sent to the UDC again. 0x05: BUSY This status is only used for Control transfer. This status is set if the data stage ends during Control transfer then a status stage token is received from the Host. When preparations for ending the status stage are complete, it ends normally and the Status Register returns to the READY state. This status is not used by the Bulk Transfer mode or the Interrupt Transfer mode. 0x06: STALL This status indicates that the applicable endpoint is in the STALL state. When in this state, the SETUP token is removed and a STALL handshake is returned. With a Control endpoint, the Status Register returns to the READY state from the STALL state when a SETUP token is received. With other endpoints, the Status Register returns to the READY state when the Initialize FIFO command is received. 0x07: INVALID This status indicates the state in which no applicable endpoint has been configured. When in this state, tokens from the Host are ignored. All endpoints are in this state when the BRESET Input signal initiates initialization. Only endpoint0 returns to the READY state when a USB reset is received. The applicable endpoint enters the READY state when configured.

Figure 8.2.41 EPx_STATUS Register (2/3)

Bits	Mnemonic	Field Name	Description
D1	FIFO_ DISABLE	FIFO_DISABLE	FIFO_DISABLE This bit indicates the state of a FIFO other than EP0. An NAK handshake is forcibly returned to all transfers if the FIFO is disabled. Access the Command Register to disable or enable the FIFO. This bit is initialized to the Enable setting when the transfer mode is changed. 0: Enable FIFO 1: Disable FIFO
D0	STAGE_ERR	STAGE_ERROR	STAGE_ERROR This bit indicates that the status stage did not end normally. This bit is set when a new SETUP token is received and the status stage has not ended normally. When this bit is “1”, it is cleared by accessing the EP0_STATUS Register from the outside. If the CPU process ends and the EOP Register is accessed, the UDC transitions to the status stage and waits for the status stage to end. If it is necessary to signal by way of a CPU application that the status change ended normally, then accessing this register when a new request flag is received makes it possible to know whether the previous request ended normally. In addition, monitoring this bit if a new flag is asserted at the application side while a request is being processed makes it possible to know if the previous request was interrupted before being completed. 0: SUCCESS 1: ERROR

Figure 8.2.41 EPx_STATUS Register (3/3)

This register is independently prepared for each endpoint. This register can be used to check the status inside the UDC from outside the UDC. Bit 5 is common to all endpoints and is the same value for all endpoints.

8.2.3.15 EPx_SIZE Register (0x260-0x2CB)

This register contains the following functions.

- During reception, this register displays the data count for one packet that was received normally at the applicable endpoint.
- During transmission, this register displays the payload size. However, when transmitting short packets, this register displays the length of those packets. It is not necessary to access this register during transmission.
- This register displays the Dual Packet mode setting and the current valid packets.

This register is available for each endpoint. There are two registers: the HIGH register that expresses upper bits 9:7 of the data size and the LOW register that consists of lower bits 6:0 and the FIFO Control bit. Since this register also supports the Dual packet mode, there are two sets of packets A and B each for the HIGH register and the LOW register. The default of registers that are initialized when the USB resets becomes the maximum payload size.

EP0_SIZE_L_A Register (Endpoint0) (0x260)

	D7	D6	D5	D4	D3	D2	D1	D0	
	PKT_ ACTIVE	DATASIZE[6:0]							
	RO	RO							: Type
	—	—							: Initial value

EP1_SIZE_L_A Register (Endpoint1) (0x264)

	D7	D6	D5	D4	D3	D2	D1	D0	
	PKT_ ACTIVE	DATASIZE[6:0]							
	RO	RO							: Type
	—	—							: Initial value

EP2_SIZE_L_A Register (Endpoint2) (0x268)

	D7	D6	D5	D4	D3	D2	D1	D0	
	PKT_ ACTIVE	DATASIZE[6:0]							
	RO	RO							: Type
	—	—							: Initial value

EP3_SIZE_L_A Register (Endpoint3) (0x26C)

	D7	D6	D5	D4	D3	D2	D1	D0	
	PKT_ ACTIVE	DATASIZE[6:0]							
	RO	RO							: Type
	—	—							: Initial value

EP1_SIZE_L_B Register (Endpoint1) (0x284)

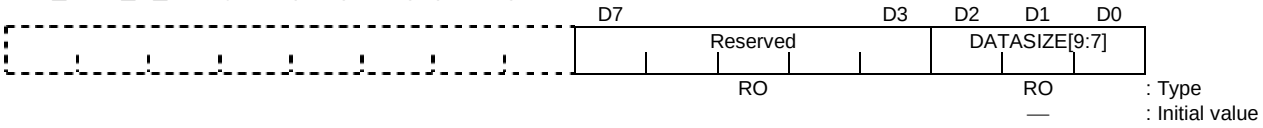
	D7	D6	D5	D4	D3	D2	D1	D0	
	PKT_ ACTIVE	DATASIZE[6:0]							
	RO	RO							: Type
	—	—							: Initial value

EP2_SIZE_L_B Register (Endpoint2) (0x288)

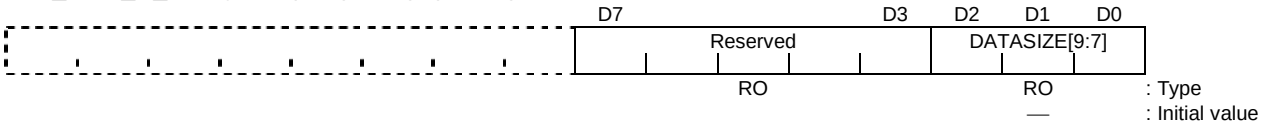
	D7	D6	D5	D4	D3	D2	D1	D0	
	PKT_ ACTIVE	DATASIZE[6:0]							
	RO	RO							: Type
	—	—							: Initial value

Bits	Mnemonic	Field Name	Description
D7	PKT_ACTIVE	PACKET ACTIVE	PACKET ACTIVE This bit specifies the packets that can be accessed when the Dual Packet mode is selected. In this case, the UDC alternately access FIFO packet A and packet B that were split in half. Refer to the PACKET ACTIVE bit before accessing the FIFO inside the UDC from the CPU. Data is read from the PACKET ACTIVE = 1 packet in the case of the receive endpoint. Always use packet A when using the Single Packet mode. This bit will have no effect in this situation. 0: OUT_DISABLE 1: OUT_ENABLE
D6:D0	DATASIZE [6:0]	DATASIZE [6:0]	DATASIZE [6:0] During reception, these bits display the data count for a single packet that the UDC received from the Host. This register is updated when reception from the Host ends normally.

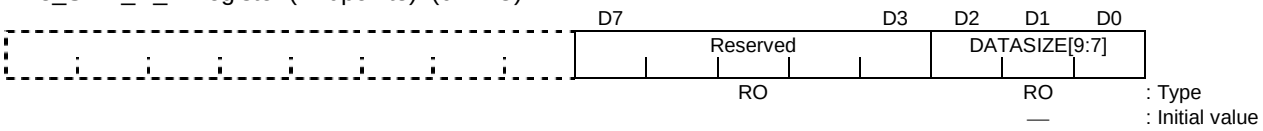
EP1_SIZE_H_A Register (Endpoint1) (0x2A4)



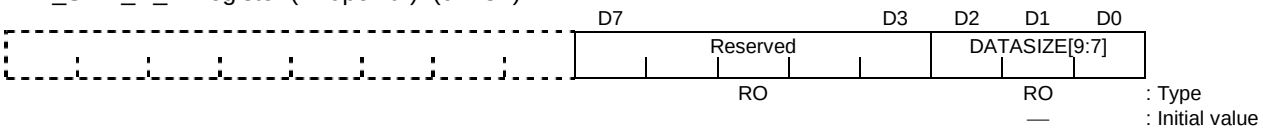
EP2_SIZE_H_A Register (Endpoint2) (0x2A8)



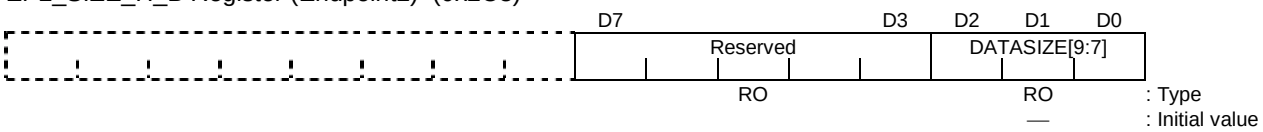
EP3_SIZE_H_A Register (Endpoint3) (0x2AC)



EP1_SIZE_H_B Register (Endpoint1) (0x2C4)



EP2_SIZE_H_B Register (Endpoint2) (0x2C8)



Bits	Mnemonic	Field Name	Description
D7:D3		Reserved	
D2:D0	DATASIZE [9:7]	DATASIZE [9:7]	DATASIZE [9:7] During reception, this register displays the data count for a single packet that the UDC received from the Host. This register is updated when reception from the Host ends normally.

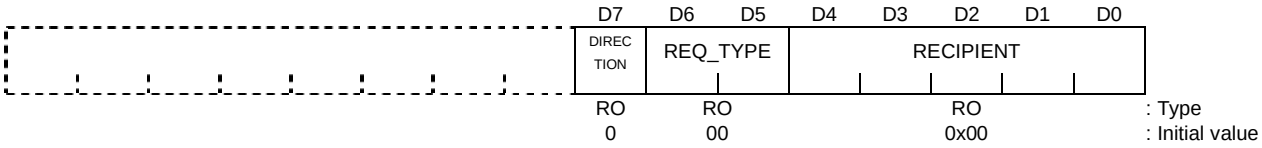
Figure 8.2.42 EPx_SIZE Register

Device requests

Device requests received from USB are stored in the following 8-byte registers. Device requests consist of the following eight registers: bmRequestType, bRequest, wValue_L, wValue_H, wIndex_L, wIndex_H, wLength_L, and wLength_H. These registers are updated each time a new SETUP token is received from the Host.

The INT_SETUP interrupt is asserted and receipt of a new device request is only signaled when data is received without any errors. In addition, there are requests that are automatically processed inside the UDC by the received request. In this case, the INT_SETUP interrupt is not externally asserted and the UDC uses STANDARD_REQUEST_FLAG and REQUEST_FLAG to signal to the outside which request is currently being processed.

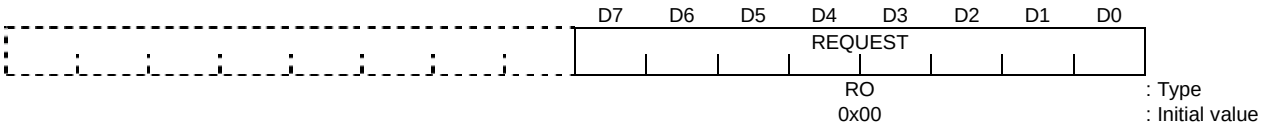
8.2.3.16 bmRequestType Register (0x300)



Bits	Mnemonic	Field Name	Description
D7	DIRECTION	DIRECTION	DIRECTION 0: OUT From the Host to the device 1: IN From the device to the Host
D6:D5	REQ_TYPE	REQ_TYPE	REQ_TYPE 0x00: Standard 0x01: Class 0x02: Vendor 0x03: Reserved
D4:D0	RECIPIENT	RECIPIENT	RECIPIENT 0x00: Device 0x01: Interface 0x02: Endpoint 0x03: Etc 0x04-0x1F: Reserved

Figure 8.2.43 bmRequestType Register

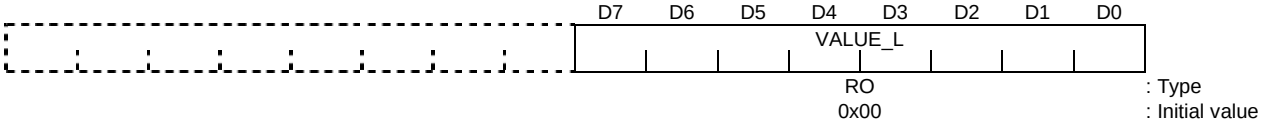
8.2.3.17 bRequest Register (0x304)



Bits	Mnemonic	Field Name	Description
D7:D0	REQUEST	REQUEST	REQUEST STANDARD 0x00: GET_STATUS 0x01: CLEAR_FEATURE 0x02: Reserved 0x03: SET_FEATURE 0x04: Reserved 0x05: SET_ADDRESS 0x06: GET_DESCRIPTOR 0x07: SET_DESCRIPTOR 0x08: GET_CONFIGURATION 0x09: SET_CONFIGURATION 0x0A: GET_INTERFACE 0x0B: SET_INTERFACE

Figure 8.2.44 bRequest Register

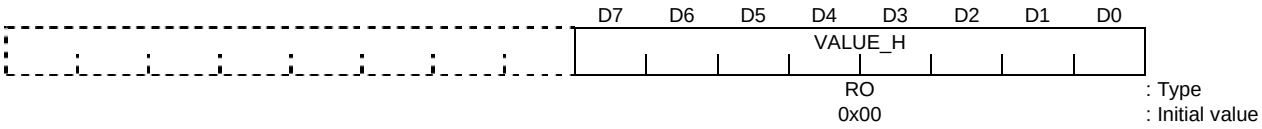
8.2.3.18 wValue_L Register (0x308)



Bits	Mnemonic	Field Name	Description
D7:D0	VALUE_L	VALUE_L [7:0]	VALUE_L [7:0]. These are the lower bytes of the field whose word size is determined by a request

Figure 8.2.45 wValue_L Register

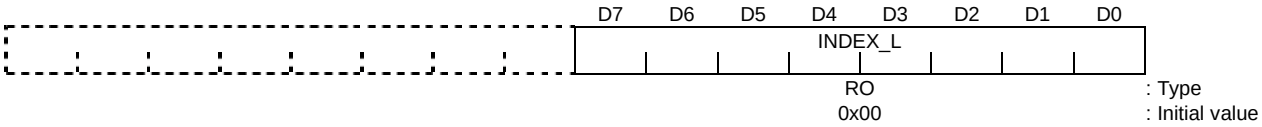
8.2.3.19 wValue_H Register (0x30C)



Bits	Mnemonic	Field Name	Description
D7:D0	VALUE_H	VALUE_H [7:0]	VALUE_H [7:0] These are the upper bytes of the field whose word size is determined by a request.

Figure 8.2.46 wValue_H Register

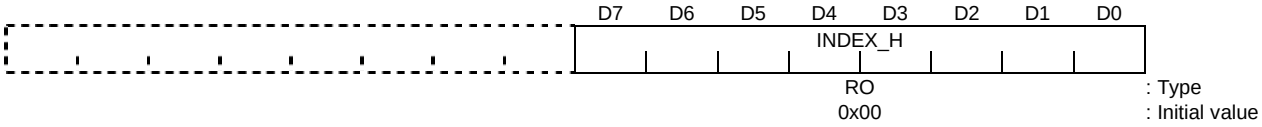
8.2.3.20 wIndex_L Register (0x310)



Bits	Mnemonic	Field Name	Description
D7:D0	INDEX_L	INDEX_L[7:0]	INDEX_L[7:0] These are the lower bytes of the field whose word size is determined by a request. This register is normally used for transferring indexes or offsets.

Figure 8.2.47 wIndex_L Register

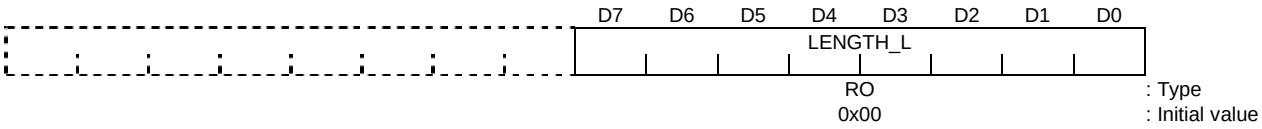
8.2.3.21 wIndex_H Register (0x314)



Bits	Mnemonic	Field Name	Description
D7:D0	INDEX_H	INDEX_H [7:0]	INDEX_H [7:0] There are the upper bytes of the field whose word size is determined by a request. This register is normally used for transferring indexes or offsets.

Figure 8.2.48 wIndex_H Register

8.2.3.22 wLength_L Register (0x318)

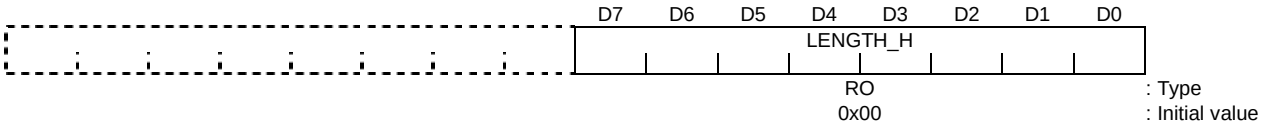


Bits	Mnemonic	Field Name	Description
D7:D0	LENGTH_L	LENGTH_L [7:0]	LENGTH_L [7:0] This register indicates the lower bytes of the transferred byte count when there is a data phase.

Figure 8.2.49 wLength_L Register

This register indicates the lower bytes of the wLength field of the device request that is transferred from the Host.

8.2.3.23 wLength_H Register (0x31C)



Bits	Mnemonic	Field Name	Description
D7:D0	LENGTH_H	LENGTH_H [7:0]	LENGTH_H [7:0] This register indicates the upper bytes of the transferred byte count when there is a data phase.

Figure 8.2.50 wLength_H Register

This register indicates the upper bytes of the wLength field of the device request that is transferred from the Host.

8.2.3.24 Setup Received Register (0x320)

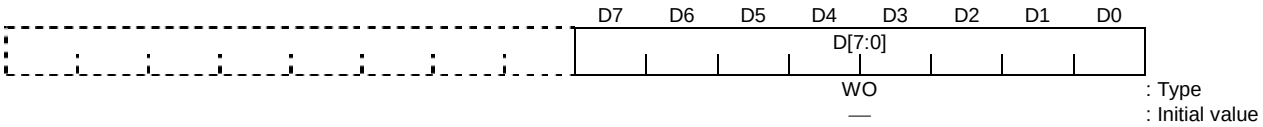


Figure 8.2.51 Setup Received Register

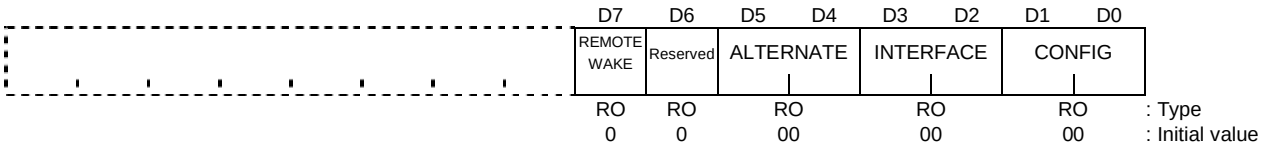
This register is used for signaling to the UDC that an external application has confirmed an INT_SETUP interrupt.

When this register is accessed from an external application, the UDC recognizes this as receipt of a device request and cancels access restriction to the EP0 FIFO. This is done to protect the data stored in EP0 if a new request is received when the previous device request has still not ended normally. The data stored in EP0 will be protected until the external application acknowledges an INT_SETUP interrupt for the new request.

However, please write 0x00 to this register when the CPU acknowledges a device request for the INT_SETUP interrupt.

After a write to this register, it takes a recovery time of two cycles at 12 MHz before the EP0_FIFO register (Section 8.2.3.12) can be accessed.

8.2.3.25 Current Config Register (0x324)



This register indicates the configuration value and interface value currently set in SET_CONFIGURATION and SET_INTERFACES.

Bits	Mnemonic	Field Name	Description
D7	REMOTE WAKE	REMOTE WAKEUP	REMOTE WAKEUP 0: Disable: Indicates that RemoteWakeup was disabled from the Host. 1: Enable: Indicates the RemoteWakeup was enabled from the Host.
D6		Reserved	
D5:D4	ALTERNATE	ALTERNATE	ALTERNATE 0x00: ALTERNATE 0 Indicates that substitute setting 0 from the Host was set. 0x01: ALTERNATE 1 Indicates that substitute setting 1 from the Host was set. 0x02: ALTERNATE 2 Indicates that substitute setting 2 from the Host was set.
D3:D2	INTERFACE	INTERFACE	INTERFACE 0x00: INTERFACE 0 Indicates that Interface 0 was set from the Host 0x01: INTERFACE 1 Indicates that Interface 1 was set from the Host 0x02: INTERFACE 2 Indicates the Interface 2 was set from the Host.
D1:D0	CONFIG	CONFIG	CONFIG 0x00: UNCONFIGURED Indicates that the data was not configured from the Host yet. 0x01: CONFIGURED 1 Indicates that Config 1 was set from the Host. 0x02: CONFIGURED 2 Indicates that Config 2 was set from the Host.

Figure 8.2.52 Current Config Register

8.2.3.26 Standard Request Register (0x328)

D7	D6	D5	D4	D3	D2	D1	D0	
S_INTERFACE	G_INTERFACE	S_CONFIG	G_CONFIG	G_DESCRIPT	S_FEATURE	C_FEATURE	G_STATUS	
RO	RO	RO	RO	RO	RO	RO	RO	: Type
0	0	0	0	0	0	0	0	: Initial value

Bits	Mnemonic	Field Name	Description
D7	S_INTERFACE	SET_INTERFACE	SET_INTERFACE
D6	G_INTERFACE	GET_INTERFACE	GET_INTERFACE
D5	S_CONFIG	SET_CONFIGURATION	SET_CONFIGURATION
D4	G_CONFIG	GET_CONFIGURATION	GET_CONFIGURATION
D3	G_DESCRIPT	GET_DESCRIPTOR	GET_DESCRIPTOR
D2	S_FEATURE	SET_FEATURE	SET_FEATURE
D1	C_FEATURE	CLEAR_FEATURE	CLEAR_FEATURE
D0	G_STATUS	GET_STATUS	GET_STATUS

Figure 8.2.53 Standard Request Register

This register indicates standard requests that are currently in progress. The bits below, when set to “1”, indicate that the corresponding request is currently in progress.

8.2.3.27 Request Register (0x32C)

	D7	D6	D5	D4	D3	D2	D1	D0	
	Reserved	Reserved	Reserved	Reserved	VENDOR	CLASS	ExSTAN DARD	STAN DARD	
	RO	RO	RO	RO	RO	RO	RO	RO	: Type
	0	0	0	0	0	0	0	0	: Initial value

Bits	Mnemonic	Field Name	Description
D7		Reserved	
D6		Reserved	SOFT_RESET
D5		Reserved	GET_PORT_STATUS
D4		Reserved	GET_DEVICE_ID
D3	VENDOR	Vendor Request	Vendor Request
D2	CLASS	Class Request	Class Request
D1	ExSTAN DARD	Standard request that does not support Auto Reply	Standard request that does not support Auto Reply (SET_DESCRIPTOR)
D0	STANDARD	Standard Request	Standard Request

Figure 8.2.54 Request Register

This register indicates device requests that are currently in progress. The bits below, when set to "1", indicate that the corresponding device request is set.

8.2.3.28 Dataset Register (0x330)

	D7	D6	D5	D4	D3	D2	D1	D0	
	Reserved	EP3_ DSET_A	EP2_ DSET_B	EP2_ DSET_A	EP1_ DSET_B	EP1_ DSET_A	Reserved	EP0_ DSET_A	
	RO	RO	RO	RO	RO	RO	RO	RO	: Type
	0	0	0	0	0	0	0	0	: Initial value

Figure 8.2.55 Dataset Register

This register indicates the presence or absence of data in the FIFO.

The external application can confirm the absence or presence of data in the FIFO at each endpoint by accessing this register. When valid transfer from the USB Host ends during data reception, the bit that corresponds to the applicable endpoint is set to “1” along with the interrupt, and then the bit is reset to “0” when the application reads out one packet of data from the FIFO. During transmission, the bit is set to “1” when transfer of one packet of data to the FIFO ends, then the bit is reset to “0” along with the interrupt when valid data is transmitted to the USB Host.

- Single Packet mode (Dataset1: D0, 2, 4, 6)

  These bits indicate that there is data in the FIFO of the applicable endpoint.

  In the case of Reception mode endpoints, data exists that should be read out to the FIFO if the bit of the applicable endpoint is in the “1” position. Access the Size Register, know the size of the data to be read, and then read that same amount of data. No data exists that should be read out if the bit is in the “0” position.

  In the case of Transmission mode endpoints, if the bit of the applicable endpoint is in the “0” position, then the CPU is able to transfer data smaller than the payload size to the FIFO. If this bit is in the “1” position, it is necessary to transfer data to the FIFO inside the UDC after the applicable bit is cleared to “0” since there is data in the FIFO waiting to be transmitted. When transmitting short packets, access the EOP Register after writing the data to be transmitted to the applicable endpoint.

- Dual Packet mode (Dataset1: D3, 5)

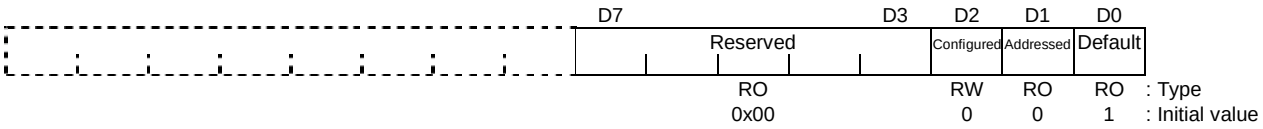
  This bit is only valid when in the Dual Packet mode. There are two packets when in the Dual Packet mode, so a DATASET bit is prepared for Packet A and Packet B. Only endpoint 1, 2 supports the Dual Packet mode.

  - When in the Reception mode, if the bits for Packets A and B of the applicable endpoint are both in the “1” position, it is necessary to confirm the PACKET_ACTIVE bit of the DATASIZE Register, know the size of the packet of data to be received, then fetch that same amount of data.

- When in the Transmission mode, if either but not both of the bits for Packets A and B of the applicable endpoint are in the “1” position, this indicates that there is space available in the Transmit FIFO. Please write data to the FIFO that is smaller than the payload size. In the case of a short packet, write the transmission data to the FIFO, access the EOP Register, then write “0” to the EOP bit of the applicable endpoint. The maximum size that can be written to one packet is equal to the MAX payload size. If both of the bits for Packets A and B are in the “0” position, then it is possible to continuously write data that is the MAX payload size × 2 bytes.

- If there is data in neither Packet A nor Packet B during transmission when in the Dual mode, the EOP Register is accessed, and “0” is written to the EOP bit of the applicable endpoint, NULL data is set in the FIFO. If the above EOP access is performed in the Single mode when there is no data in Packet A, then NULL data is set in the Packet A FIFO.

8.2.3.29 USB STATE Register (0x338)



Bits	Mnemonic	Field Name	Description
D7:D3		Reserved	
D2	Configured	Configured	Configured
D1	Addressed	Addressed	Addressed
D0	Default	Default	Default

Figure 8.2.56 USB STATE Register

This register displays the current device state during communication with the USB Host.

The Configured, Addressed, and Default bit in the above table are referred to and the reply to each device request is managed inside the UDC. When the software performs the process for a SET_CONFIG request, it is necessary to write the current state to this register. If Config 0 is specified from the Host, it becomes Unconfigged and will have to be returned to the Addressed state. Therefore, “0” must be written to bit[2] when Config 0 is specified.

The hardware automatically sets the Addressed bit (bit[1]) when “0” is written to the Configured bit (bit[2]). When the Config value that the device supports is specified from the Host, it is necessary for the device to set the mode of each endpoint using the value specified in the endpoint descriptor of that Config descriptor.

After the mode is set, please set this Configured bit (bit[2]) to “1” before accessing the EOP Register. The Addressed bit (bit[1]) is automatically reset to “0” when this bit is set to “1”.

8.2.3.30 EOP Register (0x33C)

	D7	D6	D5	D4	D3	D2	D1	D0
	Reserved	Reserved	Reserved	Reserved	EP3_EOPB	EP2_EOPB	EP1_EOPB	EP0_EOPB
					WO	WO	WO	WO : Type
					—	—	—	— : Initial value

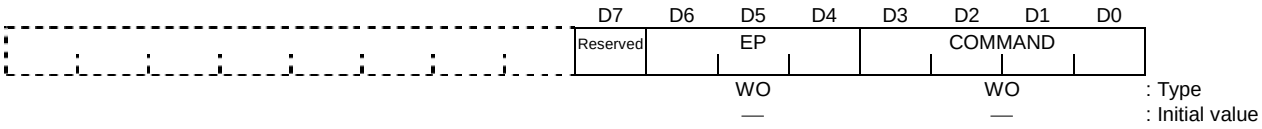
Figure 8.2.57 EOP Register

This register is used either when indicating that the control transfer data phase has ended or when transmitting a short packet during Bulk IN or Interrupt IN transfer.

During the Control transfer data phase, please write “0” to the EP0_EOPB bit either after all data to be transmitted was written to the FIFO or when all data to be received was read. The UDC uses this signal to change to the Status Stage End state.

When transmitting short packets during Bulk IN or Interrupt IN Endpoint transfer, this register is used as the write end signal for the transmitted data. Please write “0” to the EOPB bit that corresponds to the written endpoint. Please write “1” to unnecessary bits. Please input an L pulse to the EPx_EOPB signal of the external pin when performing local access via DMA.

8.2.3.31 Command Register (0x340)



Bits	Mnemonic	Field Name	Description
D7		Reserved	
D6:D4	EP	EP	EP 0x00: Selects endpoint 0 0x01: Selects endpoint 1 0x02: Selects endpoint 2 0x03: Selects endpoint 3
D3:D0	COMMAND	COMMAND	COMMAND 0x00: Reserved 0x01: Reserved 0x02: SET_DATA0 (EP0-3) This command clears the Toggle Sequence bit of the applicable endpoint. The Toggle Sequence bit of the applicable endpoint is forcibly set to “0” when this command is input. The UDC automatically performs data toggle updating by way of transfer, but it is necessary to execute this command when the Toggle Sequence bit of the endpoint is forcibly returned to “0”. In the case of Control transfer, it is not necessary to execute this command since the hardware controls everything. 0x03: RESET (EP0-3) This command resets the applicable endpoint. The applicable endpoint is initialized when this command is input. Please execute this command in order to use a CLEAR_FEATURE request to clear and endpoint STALL. (This has no effect on the Transfer mode.) This command initializes the following items. Clears Toggle Sequence bit of the applicable endpoint. Clears STALL of the applicable endpoint. Sets the FIFO_ENABLE state. 0x04: STALL (EP0-3) This command stalls the applicable endpoint. Execute this command when it is necessary to return a STALL handshake as a reply to the device request. 0x05: INVALID (EP1-3) This command sets the applicable endpoint to the Invalid state. The UDC automatically sets all endpoints except for endpoint 0 to the Invalid state when it detects a USB_RESET signal from the USB Host. If Config or Interface is changed by a device request, then it is necessary to put the unused endpoints in the Invalid state. 0x06: Reserved 0x07: FIFO_DISABLE (EP1-3) This command disables the FIFO of the applicable endpoint. When this command is set from an external device, NAK is returned to all transfers to the applicable endpoint. This command becomes effective from the next token if a packet is currently being received when this command is set from an external device. This command does not affect a packet that is currently in the process of being transferred. 0x08: FIFO_ENABLE (EP1-3) This commands enable the FIFO of the applicable endpoint. This command is used for canceling the Disable state when the FIFO_DISABLE command is used to disable the FIFO. This command also becomes effective from the next token if a packet is currently being received. The applicable endpoint enters the FIFO_ENABLE state if a USB_RESET from the Host is detected, a SET_CONFIG or SET_INTERFACE request sets the Transfer mode, or when a RESET command is executed.

Figure 8.2.58 Command Register (1/2)

Bits	Mnemonic	Field Name	Description
D3:D0	COMMAND	COMMAND	0x09INIT_DESCRIPTOR (EP0) This command is used when rewriting Descriptor RAM while the system is in operation. When the UDC detects a USB_RESET from the Host Controller, it automatically reads the contents of the Descriptor RAM, then makes all settings. If Descriptor RAM is changed while the system is in operation, please execute this command since it is necessary to read the settings again. This command does not have to be executed since the Read operation automatically starts when the USB Host is connected to. 0x0A: FIFO_CLEAR (EP1-3) This command initializes the FIFO of the applicable endpoint. However, TOGGLE is not initialized. Please execute this command when resetting by the software. This command initializes the following items. Clears STALL of the applicable endpoint. Sets the FIFO_ENABLE state. 0x0B: STALL_CLEAR (EP1-3) This command clears STALL of the applicable endpoint. Please execute this command to clear only the endpoint STALL.

Figure 8.2.58 Command Register (2/2)

This register sets commands for each end point. It is possible to use bits 6:4 to select the endpoint and bits 3:0 to set the command type. Commands that are issued for endpoints that are not supported are ignored.

Note: When writing to this register, a recovery time of five12 MHz clocks is required.

8.2.3.32 EPx_Single Register (0x344)

D7	D6	D5	D4	D3	D2	D1	D0	
Reserved	EP2_ SELECT	EP1_ SELECT	Reserved		EP2_ SINGLE	EP1_ SINGLE	Reserved	
	RW	RW			RW	RW		: Type
0	0	0	00		0	0	0	: Initial value

Bits	Mnemonic	Field Name	Description
D6	EP2_ SELECT	EP2_SELECT	EP2_SELECT
D5	EP1_ SELECT	EP1_SELECT	EP1_SELECT
D4:D3		Reserved	
D2	EP2_SINGLE	EP2_SINGLE	EP2_SINGLE
D1	EP1_SINGLE	Reserved	Reserved
D0		Reserved	

Figure 8.2.59 EPx_Single Register

This register sets the FIFO mode (SINGLE/DUAL) of each endpoint FIFO.

The EPx_SINGLE bit makes the following content valid when the EPx_SELECT bit is “1”.

0: Dual Mode

1: Single Mode

Set the EPx_SELECT bit to “1” when making the contents of the EPx_SINGLE bit valid.

0: Invalid

1: Valid

8.2.3.33 EPx_BCS Register (0x34C)

	D7	D6	D5	D4	D3	D2	D1	D0	
	Reserved	EP2_ SELECT	EP1_ SELECT	Reserved		EP2_BCS	EP1_BCS	Reserved	
		RW	RW			RW	RW		: Type
	0	0	0	00		0	0	0	: Initial value

Bits	Mnemonic	Field Name	Description
D7		Reserved	
D6	EP2_ SELECT	EP2_SELECT	EP2_SELECT
D5	EP1_ SELECT	EP1_SELECT	EP1_SELECT
D4:D3		Reserved	
D2	EP2_BCS	EP2_BCS	EP2_BCS
D1	EP1_BCS	EP1_BCS	EP1_BCS
D0		Reserved	

Figure 8.2.60 EPx_BCS Register

This register sets the access mode (CPU/DMA) to each endpoint FIFO.

The EPx_BCS bit makes the following content valid when the EPx_SELECT bit is “1”.

0: DMA access

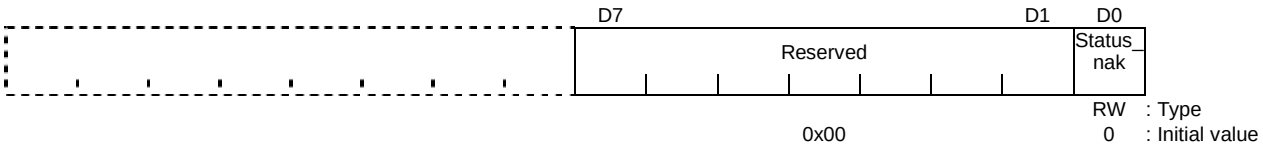
1: CPU access

Set the EPx_SELECT bit to “1” when making the contents of the EPx_BCS bit valid.

0: Invalid

1: Valid

8.2.3.34 INT_Control Register (0x358)



Bits	Mnemonic	Field Name	Description
D7:D1		Reserved	
D0	Status_nak	STATUS_NAK	STATUS_NAK 0: Disable INT_STATUS_NAK interrupts 1: Enable INT_STATUS_NAK interrupts

Figure 8.2.61 INT_Control Register

The value written to this register disables and enables the INT_STATUS_NAK interrupt. This interrupt is initialized to Disable by the BRESET signal. It is also disabled when a setup packet is received.

The INT_STATUS_NAK interrupt was added to signal a shift to the status stage. This addition was made since synchronization between the device side and the stage management gets lost if the Host completes the data phase with data that is shorter than the data length specified by wLength. When necessary, please enable this interrupt after receiving a setup packet.

8.2.3.35 Standard Request Mode Register (0x360)

	D7	D6	D5	D4	D3	D2	D1	D0	
	S_Interface	G_Interface	S_Config	G_Config	G_Descript	S_Feature	C_Feature	G_Status	
	RW	RW	RW	RW	RW	RW	RW	RW	: Type
	0	0	0	0	0	0	0	0	: Initial value

Bits	Mnemonic	Field Name	Description
D7	S_Interface	SET_ INTERFACE	SET_INTERFACE
D6	G_Interface	GET_ INTERFACE	GET_INTERFACE
D5	S_Config	SET_CONFIGU RATION	SET_CONFIGURATION
D4	G_Config	GET_CONFIGU RATION	GET_CONFIGURATION
D3	G_Descript	GET_DESCRIP TOR	GET_DESCRIPTOR
D2	S_Feature	SET_FEATURE	SET_FEATURE
D1	C_Feature	CLEAR_ FEATURE	CLEAR_FEATURE
D0	G_Status	GET_STATUS	GET_STATUS

Figure 8.2.62 Standard Request Mode Register

This register sets whether to automatically control the replies to standard requests by the hardware or by the software. Each bit expresses the request type.

Resetting the target bit to “0” sets auto reply by the hardware. Setting the target bit to “1” sets control by the software. When a hardware control request is received, disable the interrupt signal (INT_SETUP, INT_ENDPOINT0, INT_STATUS, INT_STATUSNAK). When a software control request is received, assert the interrupt signal to an external device then transfer control to the software.

8.2.3.36 Request Mode Register (0x364)

	D7	D6	D5	D4	D3	D2	D1	D0	
	Reserved	Soft_ Reset	G_Port_ Sts	G_ DeviceId	Reserved				
		RO	RO	RO					: Type
	0	0	0	0	0x0				: Initial value

Figure 8.2.63 Request Mode Register

This register sets whether to automatically reply to class requests using the hardware or to control them using the software. Each bit in this register expresses the request type.

In this device, bit[4], [5] and [6] in the Request Mode register can not be written. That is the class requests require software control. When a request to be controlled by the software is received, the interrupt signal is asserted to an external device and control changes to the software.

- SET_ADDRESS requests are only supported during auto reply.
- SET_DESCRIPTOR requests are only controlled by the software.
- Class requests include vendor requests and printer class requests are also only controlled by the software.
- Switch between auto reply and software control before connecting to the USB Host after reset. The following will happen if auto reply or software control is switched during communication with the USB Host.
  - Auto reply → Software control

    If a request is currently being received, control will switch to software control starting from the next request.
  - Software control → auto reply

    Please restrict this change since a malfunction may occur.

Note: INT_SETUP, ENDPOINT0, STATUS, STATUSNAK interrupts are only asserted by software control.

8.2.3.37 Port Status Register (0x380)

	D7	D6	D5	D4	D3	D2	D1	D0	
	Reserved	Reserved	Paper Error	Select	NotError	Reserved	Reserved	Reserved	
	WO	WO	WO	WO	WO	WO	WO	WO	: Type
	0	0	0	1	1	0	0	0	: Initial value

Figure 8.2.64 Port Status Register

This register is used when receiving requests that support the printer class. A reply containing the data stored in this register is automatically sent when a GET_PORT_STATUS request is received.

Data must be set before the request signal is received. The following bit configuration supports printer class requests. All bits in this register can be written to, so please write “0” to the reserved bits. The BRESET signal initializes this register to 0x18.

8.2.3.38 ADDRESS Register (0x38C)

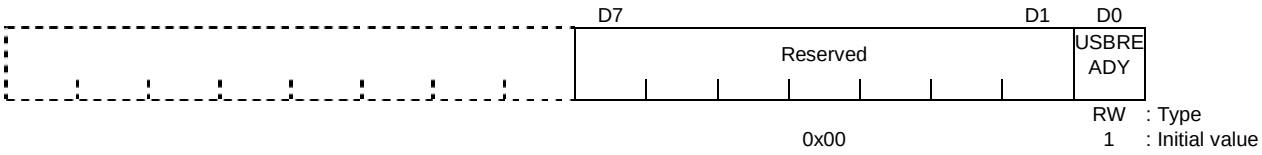
	D7	D6	D5	D4	D3	D2	D1	D0	
	Reserved	A[6:0]							
	RO				RO				: Type
	0				0x00				: Initial value

Bits	Mnemonic	Field Name	Description
D7		Reserved	
D6:D0	A[6:0]	ADDRESS[6:0]	ADDRESS [6:0] The UDC compares the addresses inside all packet IDs with the contents of this register and judges whether a transaction is valid or not. This register is set to 0x00 after a USB reset is performed.

Figure 8.2.65 ADDRESS Register

This register sets the device address specified by the USB Host during enumeration. Reading this register from outside the UDC makes it possible to confirm the current address setting.

8.2.3.39 USBREADY Register (0x398)

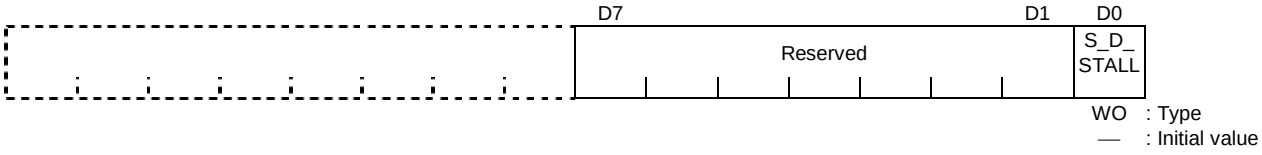


Bits	Mnemonic	Field Name	Description
D7:D1		Reser ved	
D0	USBREADY	USBREADY	USBREADY 0: Indicates that writing to the Descriptor RAM is complete. 1: Indicates that writing to the Descriptor RAM is possible. (However, do not write to the Descriptor RAM when connected to the Host.) Please add a sequence that detects the VDD signal level from the USB cable then initializes it. The UDC at this time will disable detection of the USB_RESET signal until "0" is written to the USBREADY Register after USBRESET is cleared. This prevents USB_RESET from being performed until "0" is written to the USBREADY register when the control signal is controlling the pull-up resistance on the DP signal. This is done since this signal is made equivalent to the USB_RESET signal by the Host side pull-down resistance when the Host is connected to and the pull-up resistance is in the Off state.

Figure 8.2.66 USBREADY Register

This register signals the UDC that data writing to the Descriptor RAM is complete. Always write "0" to bit 0 after the data is stored in the Descriptor RAM.

8.2.3.40 Set Descriptor STALL Register (0x3A0)



Bits	Mnemonic	Field Name	Description
D7:D1		Reserved	
D0	S_D_STALL	S_D_STALL	S_D_STALL 0: Software control (default) 1: Automatically STALL

Figure 8.2.67 Set Descriptor STALL Register

This register sets whether to automatically return STALL to a Set Descriptor request by the Data stage or the Status Stage.

8.2.3.41 Descriptor_RAM (0x800-0x9FF)

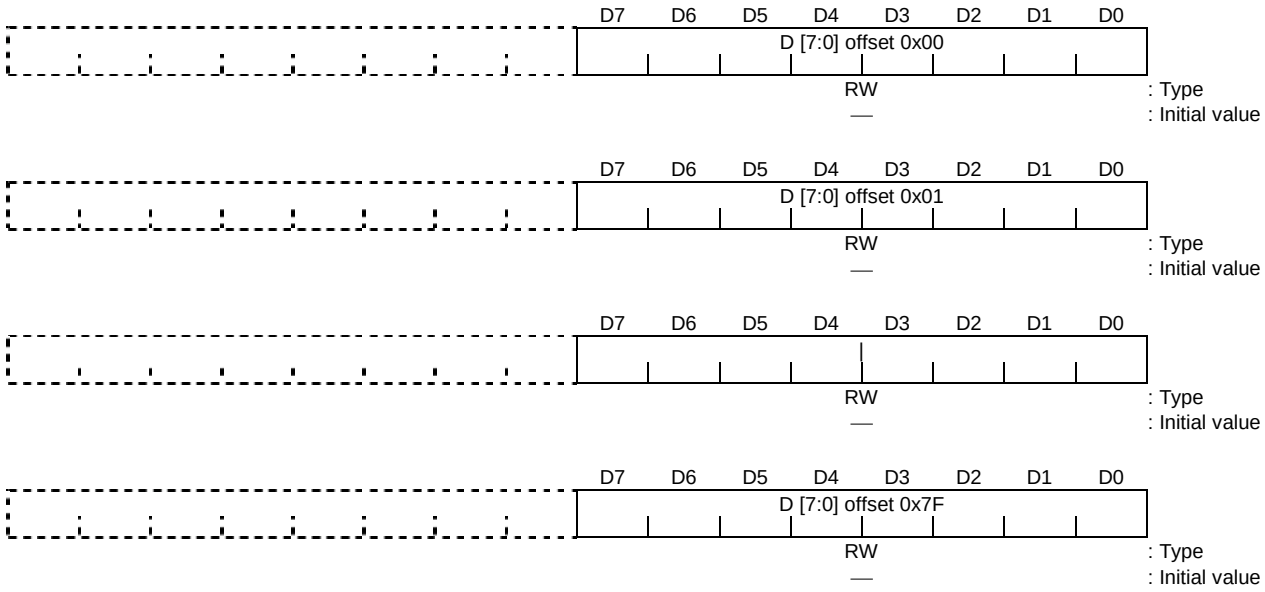


Figure 8.2.68 Descriptor_RAM

This register is used when storing a Descriptor in RAM memory. The maximum size of a Descriptor that can be stored in RAM is 128 bytes. However, it is necessary to write the Descriptor in a format that follows the Descriptor RAM configuration.

Reading or writing is only possible before USB_RESET is detected or when a SET_DESCRIPTOR request is being processed. This SET_DESCRIPTOR request process is in progress from when INT_SETUP is asserted until when the EOP Register is accessed. Please process the request according to the following sequence if there is a request to rewrite the Descriptor using SET_DESCRIPTOR.

(1) Read the Descriptor that was transferred by the SET_DESCRIPTOR request for each packet.

(2) Write all Descriptors to Descriptor RAM when reading of the last packet Descriptor is complete.

(3) Execute INIT_DESCRIPTOR of the Command Register after writing is complete.

(4) After the above process is complete, access the EOP Register and end the Status stage.

(5) This register expresses normal completion of the Status stage after an INT_STATUS interrupt is received.

Reading automatically starts when USB_RESET is detected, so it is not necessary to execute the INIT_DESCRIPTOR command when connecting to the Host.

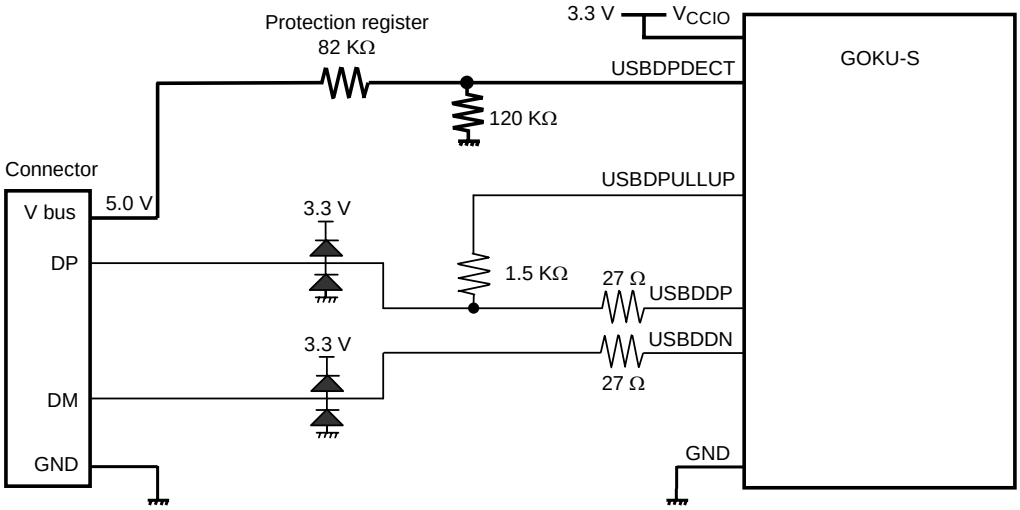
Descriptor RAM

This area is used for storing Descriptors that are defined by the USB Specifications. It is necessary to set the device, configuration, interfaces, endpoints, and string Descriptor in RAM according to the following format.

Device Descriptor	18 bytes
Config1 Descriptor (Interfaces, Endpoints)	255 bytes or less
Config2 Descriptor (Interfaces, Endpoints)	255 bytes or less
String0 Length	1 byte
String1 Length	1 byte
String2 Length	1 byte
String3 Length	1 byte
String0 Descriptor	63 bytes or less
String1 Descriptor	63 bytes or less
String2 Descriptor	63 bytes or less
String3 Descriptor	63 bytes or less

- When String Descriptors are not supported, please set the String × Length area to a size of 0. STALL is returned to String Descriptors that are not supported.
- Refer to a Descriptor string in the description that is in the Config Descriptor.
- The sequencer in the UDC determines the Config count, interface count, and endpoint count, so please pack the addresses and allocate endpoints if there are few supported endpoints.

8.3 USB DEVICE Connection Example



9. I2C Bus/SIO/GPIO Controller (Function 3)

9.1 Function Description

The GPIO (general-purpose parallel I/O port), SIO (serial I/O port) and I2C are Multiplex each pins. Their functions can select by Mode Select Resister in the GPIO module.

There have 7 modes:

- Mode0 (GPIO 5pin)
- Mode1 (I2C 1ch + GPIO 3pin)
- Mode2 (SIO Internal clock 1ch + GPIO 1pin)
- Mode3 (I2C 1ch + SIO Internal clock 1ch + GPIO 1pin)
- Mode4 (I2C 1ch + SIO External clock 1ch)
- Mode5 (SIO Internal clock 1ch + GPIO 1pin)
- Mode6 (SIO External clock 1ch)

This controller category is a PCI other communication device (Base Class 0x07, Sub-Class 0x80). It has one PCI configuration space, space for I2C I/O registers, space for SIO I/O registers, space for GPIO I/O registers and space for Internal G-bus Arbiter I/O.

9.1.1 I2C Bus Functions

9.1.1.1 Feature

The TC86C001FG (GOKU-S) contains the I2C Bus Interface channel. The I2C Bus interface is connected to external devices via two pins, SDA and SCL.

- Master Transmission/Reception
- Slave Transmission/Reception
- 7-bit addressing format
- SCL 8.1 kHz – 396.8 kHz
- General Call detection

9.1.1.2 Block Diagram

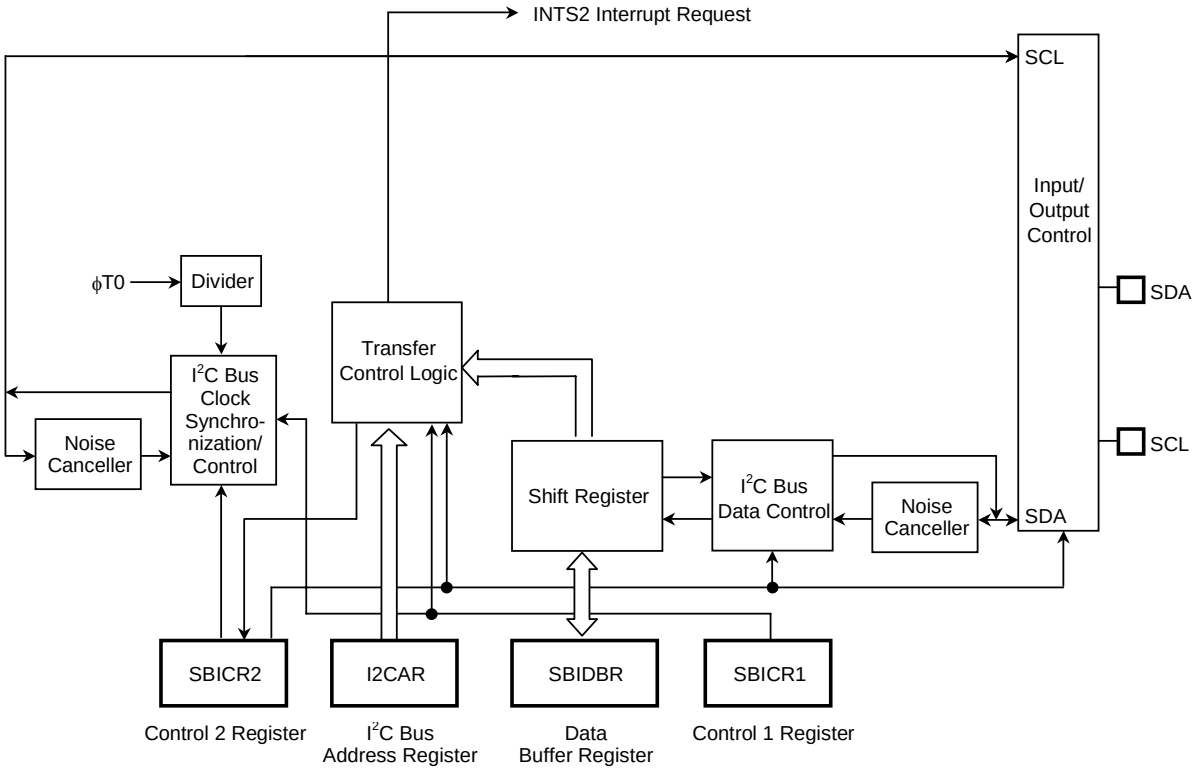


Figure 9.1.1 I²C BUS Interface Block Diagram

9.1.1.3 Registers

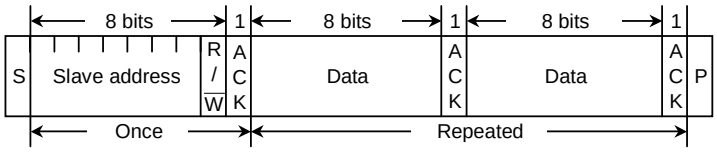
A listing of the registers used to control the I²C BUS Interface follows:

- Control 1 Register (SBICR1)
- Control 2 Register (SBICR2)
- Data Register (SBIDBR)
- I²C Bus Address Register (I2CAR)

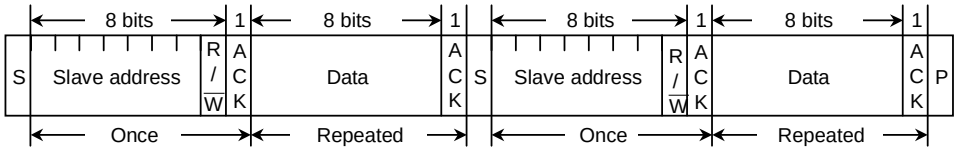
9.1.1.4 Formats

Figure 9.1.2 shows the serial bus interface data formats.

(a) Addressing format



(b) Addressing format (with repeated START condition)



(c) Free data format (master-transmitter to slave-receiver)

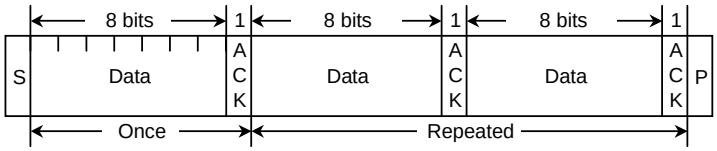


Figure 9.1.2 Data Formats

Note: S = START condition

R/$\overline{W}$ = Direction bit

ACK = Acknowledge bit

P = STOP condition

9.1.1.5 Operation

9.1.1.5.1 Acknowledgment Mode

Setting the SBICR1.ACK bit selects acknowledge mode. When operating as a master, the I²C BUS Interface sends an acknowledge clock pulse automatically after each data. As a transmitter, the I²C BUS Interface releases the SDA line during the acknowledge cycle so that the receiver of the data transfer can drive the SDA line low during this cycle to acknowledge receipt of the data. As a receiver, the I²C BUS Interface pulls the SDA line low during the acknowledge cycle after each data has been received.

Clearing the SBICR1.ACK selects non-acknowledge mode. When operating as a master, the I²C BUS Interface does not generate acknowledge clock pulses.

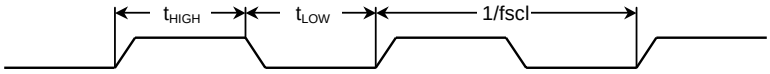
9.1.1.5.2 Number of Bits Per Transfer

The SBICR1.BC[2:0] field specifies the number of bits of the next data item to be transmitted or received. Set only 000 to BC[2:0] for the number of the next data item. Do not set to BC[2:0] except 000 for the number of the next data item. After a reset, this field is cleared to 000, causing a 7-bit slave address and the data direction (R/$\overline{W}$) bit to be transferred in a packet of eight bits. At other times, the SBICR1.BC[2:0] field keeps a previously programmed value.

9.1.1.5.3 Serial Clock

(1) I²C Bus Clock Source

The SBICR1.SCK[2:0] field controls the maximum frequency of the SCL clock driven out on the SCL pin in master mode, as illustrated below.



$T_{LOW} = 2n/\phi T0$

$T_{HIGH} = 2n/\phi T0 + 10/\phi T0$

$Fscl = 1/(t_{Low} + t_{HIGH})$

$= \dfrac{\phi T0}{2^n + 10}$

$\phi T0 = fperiph/4$ (fperiph = 66.67 MHz)

SBICR1.SCK[2:0]	N
000	5
001	6
010	7
011	8
100	9
101	10
110	11

Figure 9.1.3 I²C Bus Clock Source

(2) Clock Synchronization

Clock synchronization is performed using the wired-AND connection of all I²C-bus components to the bus. If two or more masters try to transfer messages on the I²C bus, the first to pull its clock line low wins the arbitration, overriding other masters producing a high on their clock lines.

Clock signals of two or more devices on the I²C-bus are synchronized to ensure correct data transfers. Figure 9.1.4 shows a depiction of the clock synchronization mechanism for the I²C bus with two masters.

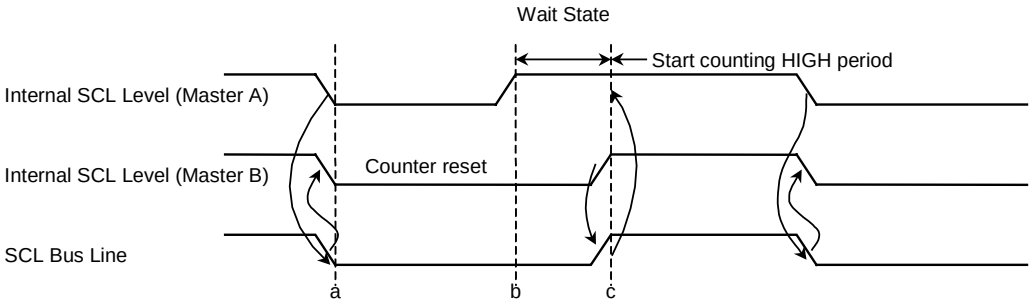


Figure 9.1.4 Clock Synchronization Example

At point a, Master A pulls its internal SCL level low, bringing the SCL bus line low. The high-to-low transition on the SCL bus line causes Master B to reset its high-level counter and pulls its internal SCL level low.

Master A completes its low period at point b. However, the low-to-high transition on its internal SCL level does not change the state of the SCL bus line if Master B's internal SCL level is still within its low period. Therefore, Master A enters a high wait state, where it does not start counting off its high period.

When Master B has counted off its low period at point c, its internal SCL level goes high, releasing the SCL bus line (high). There will then be no difference between the internal SCL levels and the state of the SCL bus line, and both Master A and Master B start counting off their high periods.

This way, a synchronized SCL clock is generated with its high period determined by the master with the shortest clock high period and its low period determined by the one with the longest clock low period.

(3) Slave Addressing and Address Recognition Mode

When the I²C BUS INTERFACE is configured to operate as a slave, the SA[6:0] field in the I2CAR must be loaded with the 7-bit I²C-bus address to which the I²C BUS INTERFACE is to respond. The ALS bit must be cleared for the I²C BUS INTERFACE to recognize the incoming slave address.

(4) Configuring the I²C BUS INTERFACE as a Master or a Slave

Setting the SBICR2.MST bit configures the I²C BUS INTERFACE as a master, and clearing it configures the I²C BUS INTERFACE as a slave. This bit is cleared by hardware when a STOP condition has been detected and when arbitration for the I²C bus has been lost.

(5) Configuring the I²C BUS INTERFACE as a Transmitter or a Receiver

The SBICR2.TRX bit is set or cleared by hardware to indicate whether the I²C BUS INTERFACE is configured as a transmitter or a receiver.

As a slave, the I²C BUS INTERFACE is put in either slave-receiver or slave-transmitter mode, depending on the value of the data direction (R/$\overline{W}$) bit transmitted by the master. When the I²C BUS INTERFACE is addressed as a slave, the TRX bit reflects the value of the R/$\overline{W}$ bit. The TRX bit is set or cleared on the following occasions:

- when transferring data using addressing format
- when the received slave address matches the value in I2C0CR
- when a general-call address is received; i.e., the eight bits following the START condition are all zeros.

As a master, the I²C BUS INTERFACE is put in either master-transmitter or a master-receiver mode when it has received an acknowledge from an addressed slave. The TRX bit indicates the opposite of the R/$\overline{W}$ bit sent by the I²C BUS INTERFACE. If the I²C BUS INTERFACE does not receive an acknowledge from a slave, the TRX bit retains the previous value.

The TRX bit is cleared by hardware when a STOP condition has been detected and when arbitration for the I²C bus has been lost.

(6) Generating START and STOP Conditions

When the SBICR2.BB bit is cleared, the bus is free. At this time, writing 1s to the MST, TRX, BB and PIN bits in the SBICR2 causes the I²C BUS INTERFACE to generate a START condition on the bus and shift out 8-bit I²C-bus data. Before generating a START condition, the ACK bit must be set to 1.

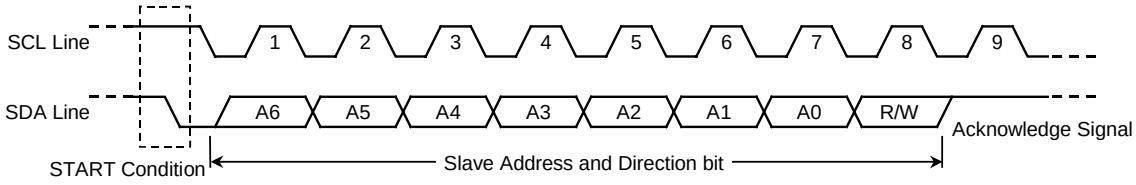


Figure 9.1.5 Generating a START Condition and a Slave Address

When the SBICR2.BB bit is set, the bus is busy. When SBICR2.BB=1, writing 1s to the MST, TRX and PIN bits and a 0 to the BB bit causes the I²C BUS INTERFACE to start a sequence for generating a STOP condition on the bus to abort the transfer. The MST, TRX, BB and PIN bits should not be altered until a STOP condition appears on the bus.

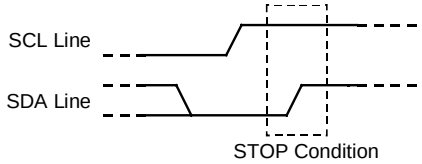


Figure 9.1.6 Generating a STOP Condition

The BB bit can be read to determine if the I^2C bus is in use. The BB bit is set when a START condition is detected and cleared when a STOP condition is detected.

(7) Asserting and Deasserting Interrupt Requests

When an I^2C BUS INTERFACE interrupt (INTS2) is generated, the Pending Interrupt Not (PIN) bit in the SBICR2 is cleared to 0. While the PIN is 0, the I^2C BUS INTERFACE pulls the SCL line low.

After transmission or reception of one data word on the I^2C bus, the PIN bit is automatically cleared. In transmitter mode, the PIN bit is subsequently set to 1 each time the SBIDBR is written. In receiver mode, the PIN bit is set to 1 each time the SBIDBR is read.

It takes a period of t_{LOW} for the SCL line to be released after the PIN bit is set.

In Address Recognition mode (ALS=0), the PIN bit is cleared when the I^2C BUS INTERFACE is addressed as a slave and the received slave address matches the value in the I2CAR or is all 0s (i.e., a general call).

A write of 1 by software sets the PIN bit, but a write of 0 has not effect on this bit.

(8) Slave Address Match Monitor

When acting as a slave-receiver, the ALS bit in the I2CAR determines whether the I^2C BUS INTERFACE recognizes the incoming slave address or not. In Address Recognition mode (i.e., ALS=0), the Addressed-As-Slave (AAS) bit in the SBICR2 is set when an incoming address over the I^2C-bus matches the value in the I2CAR or when the general-call address has been received. When ALS=1, the AAS bit is set when the first data word has been received. The AAS bit is cleared each time the SBIDBR is read or written.

(9) General-Call Detection Monitor

When acting as a slave receiver, the AD0 bit in the SBICR2 is set when a general-call address has been received. The general-call address is detected when the eight bits following a START condition are all zeros. The AD0 bit is cleared when a START or STOP condition is detected on the bus.

(10) Last Received Bit Monitor

The LRB bit in the SBICR2 holds the value of the last bit received over the SDA line at the rising edge of the SCL clock. In Acknowledgement mode, reading this bit immediately after generation of the INTS2 interrupt returns the value of the ACK signal.

(11) Software Reset

The I^2C BUS INTERFACE provides a software reset, which permits recovery from system lockups caused by external noise. A software reset is performed by a write of 10 followed by a write of 01 to the SWRST[1:0] field in the SBICR2. After a software reset, all control and status register bits are initialized to their reset values. Upon resetting the I^2C BUS INTERFACE, the SWRST[1:0] field is automatically cleared to 00.

(12) Serial Bus Interface Data Buffer Register (SBIDBR)

The SBIDBR is a data buffer interfacing to the I²C bus. All read and write operations to/from the I²C bus are done via this register.

When the I²C BUS INTERFACE is acting as a master, loading this register with a slave address and a data direction bit causes a START condition to be generated.

(13) I²C Bus Address Register (I2CAR)

When the I²C BUS INTERFACE is configured as a slave, the SA[6:0] field in the I2CAR must be loaded with the 7-bit I²C-bus address to which the I²C BUS INTERFACE is to respond.

If the ALS bit in the I2CAR is cleared, the I²C BUS INTERFACE recognizes a slave address transmitted by the master device, interpreting incoming frame structures as in addressing format. If the ALS bit is set, the I²C BUS INTERFACE does not recognize a slave address and interprets all frame structures as in free data formats.

9.1.1.6 Programming Sequences in I²C Bus Mode

(1) I²C BUS INTERFACE Initialization

First, the ACK and SCK[2:0] bits in the SBICR1. Write 0s to bit[7:5] and bit[3] in the SBICR1.

Next, program the I2CAR. The SA[6:0] field in the I2CAR defines the chip's slave address, and the ALS bit (bit[0]) selects an address recognition mode. (The ALS bit must be cleared when using the addressing format.)

Next, program the SBICR2 to initially configure the I²C BUS INTERFACE in slave-receiver mode; i.e., clear the MST, TRX and BB bits to 0, set the PIN bit to 1. Write 0s to bit[3:0] in the SBICR2.

	7	6	5	4	3	2	1	0	
SBICR1 ←	0	0	0	X	0	X	X	X	Disable generation of ACK and select SCL clock frequency.
I2CAR ←	X	X	X	X	X	X	X	X	Load a slave address and selects address recognition mode.
SBICR2 ←	0	0	0	1	0	0	0	0	Configure the I2C BUS INTERFACE in slave-receiver mode.

Note: X = Don't care

(2) Generating a START Condition and a Slave Address

1) Master Mode

In master mode, the following steps are required to generate a START condition and a slave address on the I²C-bus.

First, ensure that the bus is free (i.e., SBICR2.BB = 0).

Next, set the ACK bit in the SBICR1 to enable generation of acknowledge clock pulses. Then, loads the SBIDBR with a slave address and a data direction bit to be transmitted via the I²C-bus.

When BB=0, writing 1s to the MST, TRX, BB and PIN bits in the SBICR2 causes a START condition to be generated on the bus. Following a START condition, the I²C BUS INTERFACE generates SCL clock pulses nine times: the I²C BUS INTERFACE shifts out the contents of the SBIDBR with the first eight SCL clocks, and releases the SDA line during the last (i.e., ninth) SCL clock to receive an acknowledgement signal from the addressed slave.

The INTS2 interrupt request is generated on the falling edge of the ninth SCL clock pulse, and the PIN bit in the SBICR2 is cleared to 0. In master mode, the SBI pulls the SCL line low while the PIN bit is 0. Upon interrupt, the TRX bit either remains set or is cleared according to the value of the transmitted direction bit, provided an acknowledgement signal has been returned from the slave.

Settings in main routine

```
                  7 6 5 4 3 2 1 0
   Reg.      ← SBICR2
   Reg.      ← Reg. & 0x20
   if Reg.   ≠ 0x00                  Ensure that the bus is free.
   Then
   SBICR1    ← X X X 1 0 X X X     Select Acknowledgement mode.
   SBIDBR    ← X X X X X X X X     Load the slave address and a data direction bit.
   SBICR2    ← 1 1 1 1 0 0 0 0     Generate a START condition.
```

2) Slave Mode

In slave mode, the following steps are required to receive a START condition and a slave address via the I²C-bus.

Upon detection of a START condition, the I²C BUS INTERFACE clocks in a 7-bit slave address and a data direction bit transmitted by the master during the first eight SCL clock pulses. If the received slave address matches its own address in the I2CAR or is equal to the general-call address (0x00), the I²C BUS INTERFACE pulls the SDA line low during the last (i.e., ninth) SCL clock for acknowledgement.

The INTS2 interrupt request is generated on the falling edge of the ninth SCL clock pulse, and the PIN bit in the SBICR2 is cleared to 0. In slave mode, the I²C BUS INTERFACE pulls the SCL line low while the PIN bit is 0.

Note: The user can only use a DMA transfer:

- when there is only one master and only one slave on the I²C bus; and continuous transmission or reception is possible.

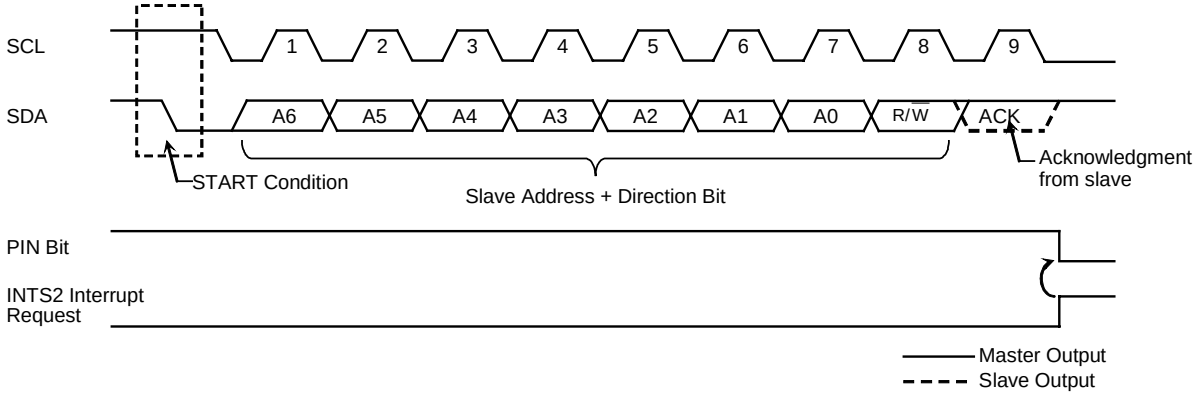


Figure 9.1.7 Generation of a START Condition and a Slave Address

(3) Transferring a Data Word

Each time a data word has been transmitted or received, the INTS2 interrupt is generated. It is the responsibility of the INTS2 handler to test the MST bit in the SBICR2 to determine whether the I²C BUS INTERFACE is in master or slave mode.

1) Master Mode (SBICR2.MST = 1)

If the MST bit in the SBICR2 is set, test the TRX bit in the same register to determine whether the I²C BUS INTERFACE is in master-transmitter or master-receiver mode.

Master-Transmitter Mode (SBICR2.TRX = 1)

Test the LRB bit in the SBICR2. If the LRB bit is set, that means the slave-receiver requires no further data to be sent from the master-transmitter. The master-transmitter must then generate a STOP condition as described later to stop transmission.

If the LRB bit is cleared, that means the slave-receiver requires further data. If the number of bits per transfer is 8, then write the transmit data into the SBIDBR. When the SBIDBR is loaded, the PIN bit in the SBICR2 is set to 1, and the transmit data is shifted out from the SDA pin, clocked by the SCL clock. Once the transfer is complete, the INTS2 interrupt is generated, the PIN bit is cleared, and the SCL line is pulled low. If further data is required, test the LRB bit again and repeat the above procedure.

INTS2 interrupt

```
if MST = 0
Then go to slave-mode processing
if TRX = 0
Then go to receiver-mode processing
if LRB = 0
Then go to processing for generating a STOP condition
SBICR1  ← X X X X 0 X X X      Set number of bits to be transmitted and specify whether
                               ACK is required.
SBIDBR  ← X X X X X X X X      Load the transmit data.
End of interrupt processing
X = Don't care
```

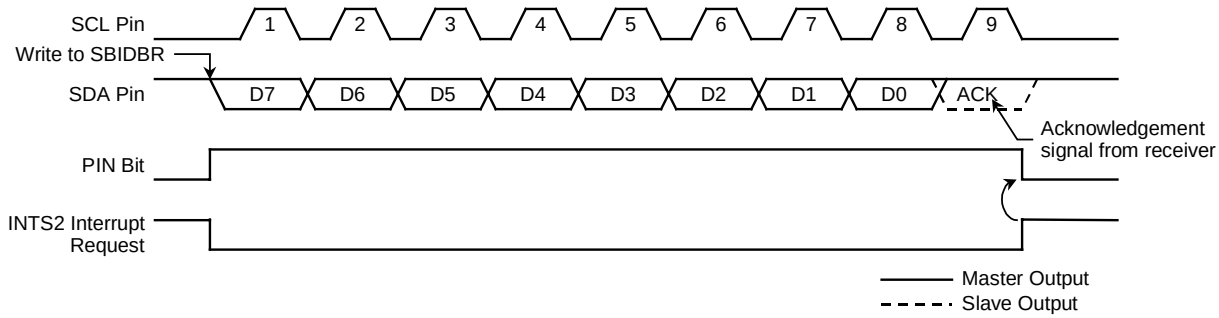


Figure 9.1.8 SBICR1.BC[2:0] = 000 and SBICR1.ACK = 1 (Master-Transmitter Mode)

Master-Receiver Mode (SBICR2.TRX = 0)

If the number of bits per transfer is 8, read the SBIDBR. The first read of the SBIDBR is a dummy read because data has not yet been received. A dummy read returns an undefined value. Upon this read, the SCL line is released, the PIN bit in the SBICR2 is set, and the SCL clock is driven out to receive a data word into the SBIDBR. The master-transmitter generates an acknowledgement signal (i.e., a low level) on the SDA line following the last received bit.

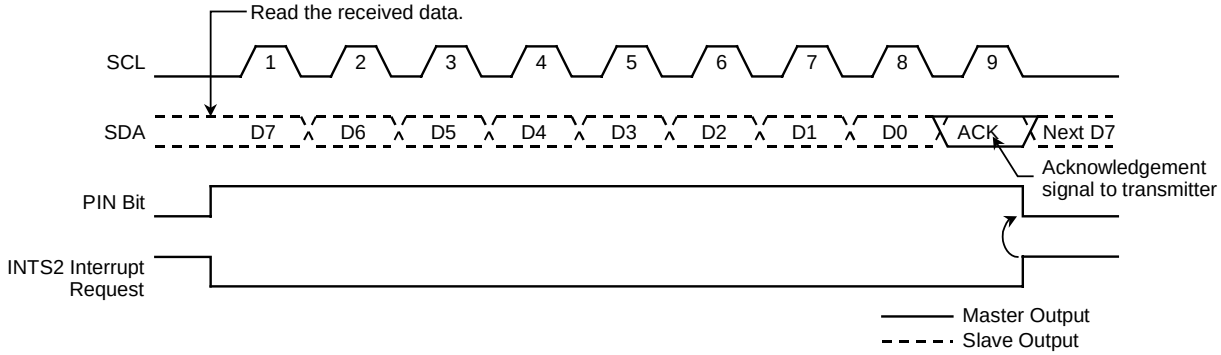


Figure 9.1.9 SBICR1.BC[2:0] = 000 and SBICR1.ACK = 1 (Master-Receiver Mode)

To prepare to terminate the data transfer, the master-receiver must clear the ACK bit in the SBICR1 immediately before the read of the second to last data word. This causes an acknowledge clock pulse not to be generated on the last data word.

When the transfer is complete, the INTS2 interrupt is generated. After interrupt processing, the INTS2 interrupt handler must set the BC[2:0] field in the SBICR1 to 001 and read the SBIDBR, so that a clock is generated on the SCL line once. With the ACK bit cleared, the master-receiver holds the SDA line high, which signals the end of transfer to the slave-transmitter.

Then, the I²C BUS INTERFACE generates the INTS2 interrupt again, whereupon the INTS2 handler must generate a STOP condition to stop communication via the I²C bus.

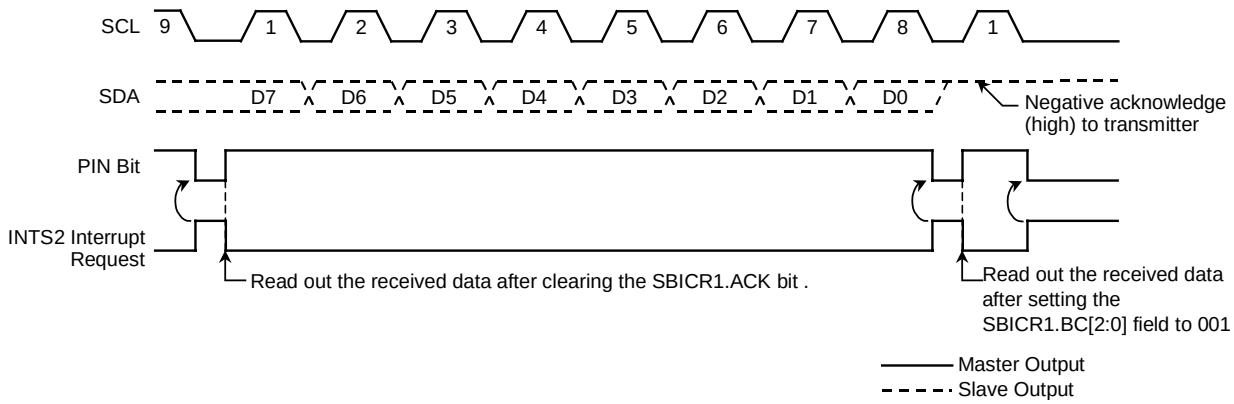


Figure 9.1.10 Terminating Data Transmission in Master-Receiver Mode

Example: When receiving N Data Words

INTS2 interrupt (after data transmission)

```
                7 6 5 4 3 2 1 0
SBICR1  ← X X X X 0 X X X      Set the number of bits to be received and specify
                                whether ACK is required.
Reg.    ← SBIDBR               Dummy read
End of interrupt
```

INTS2 interrupt (first to (N-2)th data reception)

```
                7 6 5 4 3 2 1 0
Reg.    ← SBIDBR               Read the first to (N-2)th data words.
End of interrupt
```

INTS2 interrupt ((N-1)th data reception)

```
                7 6 5 4 3 2 1 0
SBICR1  ← X X X 0 0 X X X      Disable generation of acknowledgement clock.
Reg.    ← SBIDBR               Read the (N-1)th data words.
End of interrupt
```

INTS2 interrupt (Nth data reception)

```
                7 6 5 4 3 2 1 0
SBICR1  ← 0 0 1 0 0 X X X      Generate a clock once.
Reg.    ← SBIDBR               Read the Nth data word.
End of interrupt
```

INTS2 interrupt (after completing data reception)

Processing for generating a STOP condition. Stop communication.
End of interrupt
X = Don't care

2) Slave Mode (SBICR2.MST = 0)

If the MST bit in the SBICR2 is cleared, the I²C BUS INTERFACE is in slave mode. In slave mode, the I²C BUS INTERFACE generates the INTS2 interrupt on four occasions: 1) when the I²C BUS INTERFACE has received any slave address; 2) when the I²C BUS INTERFACE has received a general-call address; 3) when the received slave address matches its own address in the I2CAR; and 4) when a data transfer has been completed in response to a general-call.

Also, if the I²C BUS INTERFACE, as a master, loses arbitration for the I²C bus, it switches to slave mode. If arbitration is lost during a data word transfer, SCL continues to be generated until the data word is complete; then the INTS2 interrupt is generated.

When the INTS2 interrupt occurs, the PIN bit in the SBICR2 is cleared, and the SCL line is pulled low. When the SBIDBR is read or written or when the PIN bit is set back to 1, the SCL line is released after a period of t_{LOW}.

Processing to be done in slave mode varies, depending on whether or not the I2C BUS INTERFACE has switched over to slave mode as a result of lost arbitration.

Test the AL, TRX, AAS and AD0 bits in the SBICR2 to determine the processing required, as summarized in Table 9.1.1.

Example: When the received slave address matches the I²C BUS INTERFACE's own address and the data direction (R/$\overline{W}$) bit is 1

INTS2 interrupt

if TRX = 0
Then go to other processing
if AAS = 0
Then go to other processing

```
SBICR1  ← X X X 1 0 X X X      Set the number of bits to be transmitted.
SBIDBR  ← X X X X 0 X X X      Load the transmit data.
```

X = Don't care

Table 9.1.1 Processing in Slave Mode

TRX	AL	AAS	AD0	State	Processing
1	0	1	0	In slave-receiver mode, the I²C BUS INTERFACE received a slave address with the direction bit set transmitted by the master.	Set 000b to the SBICR1. BC[2:0] for the number of bits in a data word and write the transmit data into the SBI0DBR.
		0	0	In slave-transmitter mode, the I²C BUS INTERFACE has completed a transmission of one data word.	Test the SBICR2.LRB bit. If the LRB bit is set, that means the master-receiver does not require further data. Set the SBICR2.PIN bit to 1 and clear the TRX bit to 0 to release the bus. If the LRB bit is cleared, that means the master-receiver requires further data. Set 000b to the SBICR1. BC[2:0] for the number of bits in the data word and write the transmit data to the SBIDBR.
0	0	1	1/0	In slave-receiver mode, the I²C BUS INTERFACE received either a slave address with the direction bit cleared or a general-call address transmitted by the master.	Read the SBIDBR (a dummy read) to set the SBICR2.PIN bit to 1, or write a 1 to this bit.
		0	1/0	In slave-receiver mode, the I²C BUS INTERFACE has completed a reception of a data word.	Set 000b to the SBICR1. BC[2:0] for the number of bits in the data word and read the received data from the SBIDBR.

(4) Generating a STOP Condition

When the SBICR2.BB bit is set, setting the MST, TRX and PIN bits in the SBICR2 to 1 and clearing the BB bit in the same register causes the I²C BUS INTERFACE to start a sequence for generating a STOP condition on the I²C bus. Do not alter the contents of these bits until the STOP condition is present on the bus.

If another device is pulling down the SCL bus line, the I²C BUS INTERFACE waits until the SCL line is released (high) again; when SCL is high, the I²C BUS INTERFACE drives the SDA pin high to generate a STOP condition.

```
              7 6 5 4 3 2 1 0
SBICR2  ← 1 1 0 1 0 0 0 0        Generate a STOP condition.
```

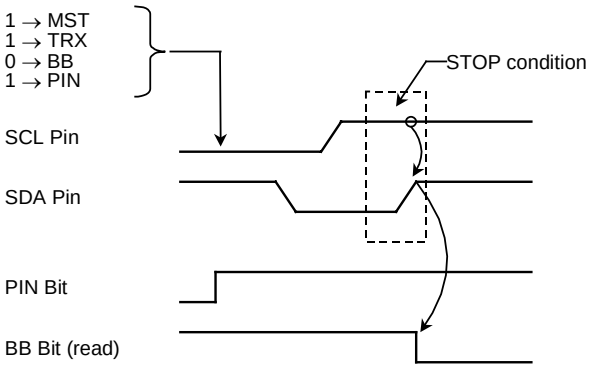


Figure 9.1.11 Generating a STOP Condition

(5) Repeated START Condition

A data transfer is always terminated by a STOP condition. However, if a master still wishes to communicate on the bus, it can generate a repeated START condition and address another slave or change the data direction without first generating a STOP condition. The following describes the steps required to generate a repeated START condition.

First, clear the MST, TRX and BB bits in the SBICR2 and set the PIN bit in the same register to release the bus. This causes the SDA pin to be held high and the SCL pin to be released. Because no STOP condition is generated on the bus, other devices think that the bus is busy.

Then, poll the SBICR2.BB bit until it is cleared to ensure that the SCL pin is released. Next, poll the LRB bit until it is set to ensure that no other device is pulling the SCL bus line low. Once the bus is determined to be free this way, use the steps described in (2) to generate a START condition.

To satisfy the minimum setup time of the START condition, in Standard-mode, at least 4.7-μs wait period must be created by software after the bus becomes free.

```
                    7 6 5 4 3 2 1 0
SBICR2  ← 0 0 0 1 0 0 0 0      Releases the bus.
if SBICR2<BB> ≠ 0              Check that the SCL pin is released.
Then
if SBICR2<LRB> ≠ 1             Check that no other device is pulling the SCL line low.
Then
4.7-μs Wait
SBICR1  ← X X X 1 0 X X X      Select Acknowledgement Mode.
SBIDBR  ← X X X X X X X X      Load a slave address and the direction bit.
SBICR2  ← 1 1 1 1 0 0 0 0      Generate a START condition.
X = Don't care
```

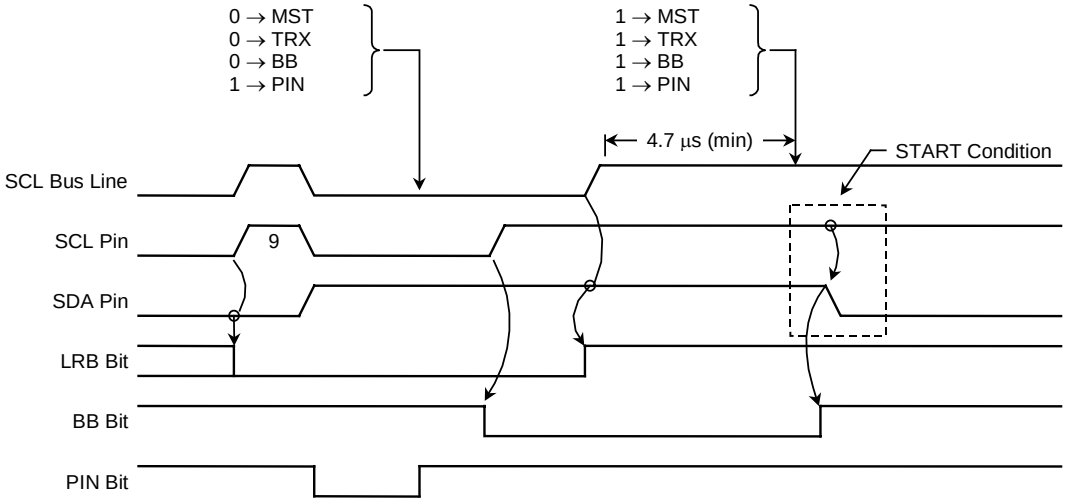


Figure 9.1.12 Repeated START Condition

9.1.2 SIO Controller Functions

9.1.2.1 Features

The GOKU-S asynchronous Serial I/O (SIO) interface has one full duplex UART channel. SIO has the following features.

(1) Full duplex transmission (simultaneous transmission and reception)

(2) On-chip baud rate generator

(3) Modem flow control (CTS*/RTS*) if Mode5 or Mode6 is selected.

(4) External Clock Input (SCLK) if Mode4 or Mode6 is selected.

(5) FIFO

- Transmit FIFO: 8 bits × 8 stages
- Reception FIFO: 13 bits × 16 stages (data: 8 bits, status: 5 bits)

(6) Supports multi-controller systems

- Supports Master/Slave operation

(7) DMA transfer is not supported

9.1.2.2 Block Diagram

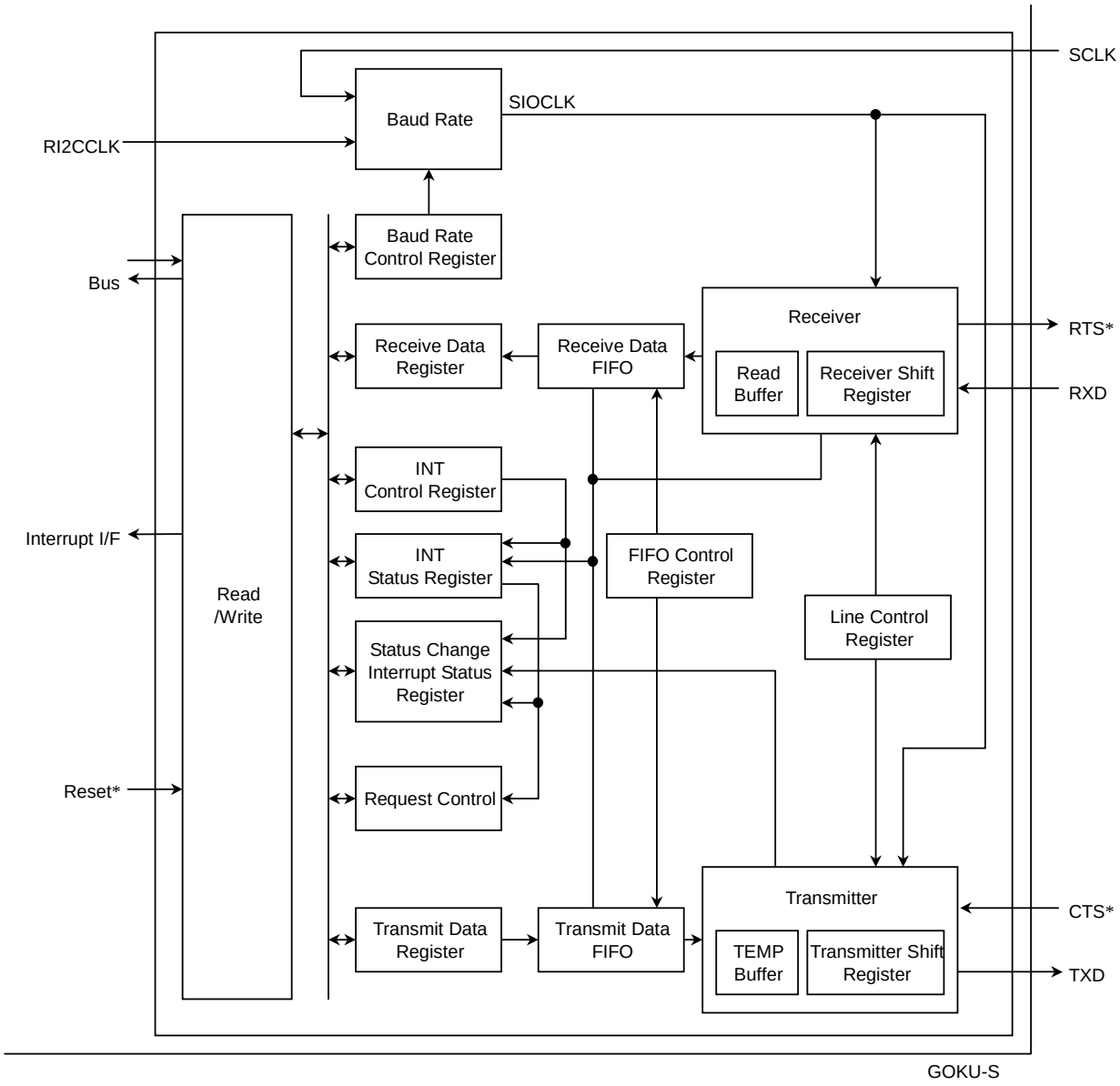


Figure 9.1.13 SIO Internal Block Diagram

9.1.2.3 Overview

During reception, serial data that are input as an RXD signal from an external source are converted into parallel data, and then are stored in the Receive FIFO buffer. Parallel data stored in the FIFO buffer are fetched by CPU.

During transmission, parallel data written to the Transmit FIFO buffer by CPU are converted into serial data, and then are output as a TXD signal.

9.1.2.4 Data Format

The GOKU-S SIO can use the following data formats.

Data Length:	8/7 bits
Stop Bit:	1/2 bits
Parity Bit:	Yes/No
Parity Format:	Even/Odd
Start Bit:	Fixed to 1 bit

Figure 9.1.14 illustrates the data frame when making each setting.

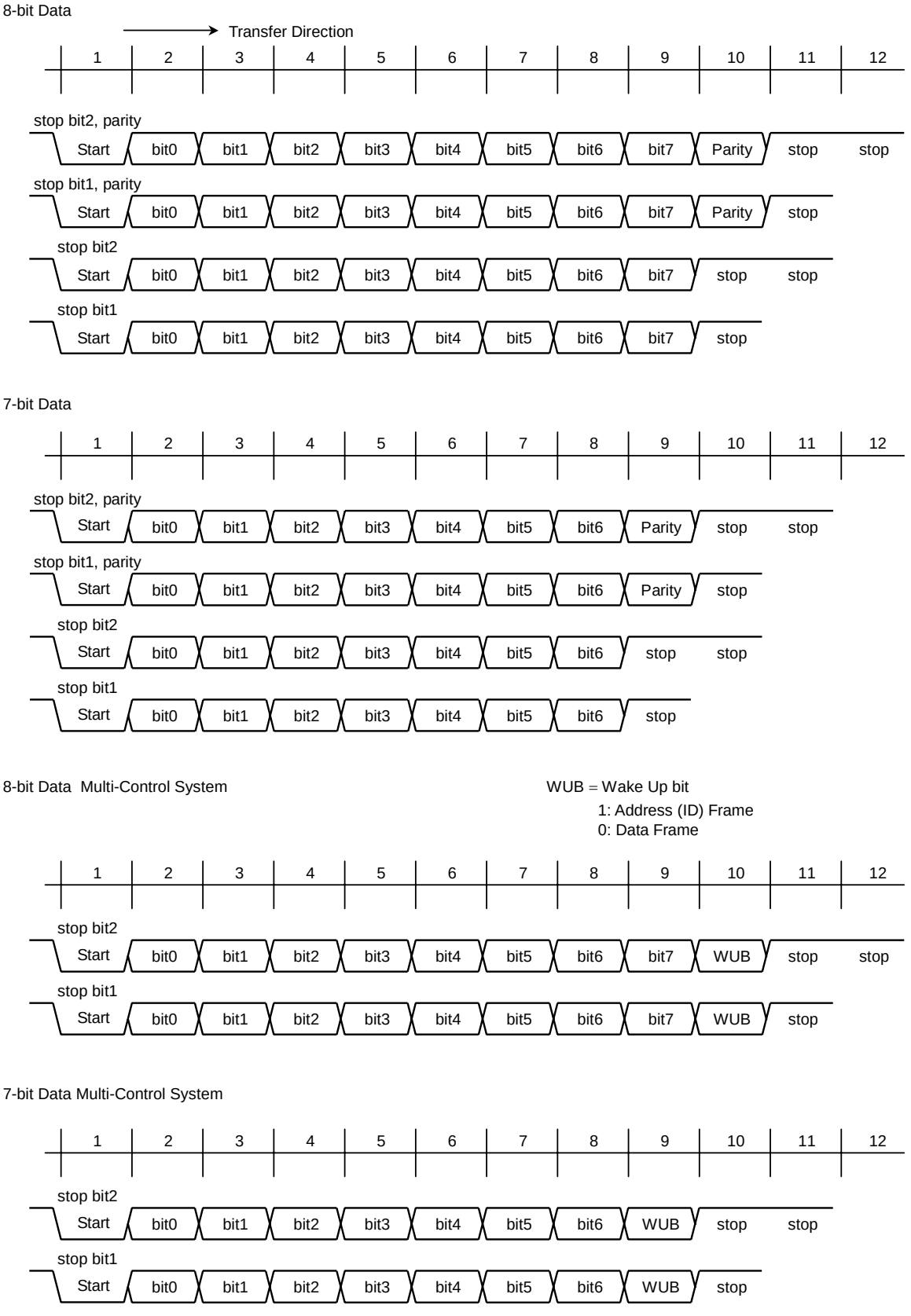


Figure 9.1.14 Data Frame Configuration

9.1.2.5 Serial Clock Generator

Generates the Serial Clock (SIOCLK). SIOCLK determines the serial transfer rate and has a frequency that is 16× the baud rate. One of the following can be selected as the source for the Serial Clock (SIOCLK).

- Internal Clock (RI2CCLK)
- External Clock Input (SCLK)
- Baud rate generator circuit output

The RI2CCLK frequency is 66.67 MHz. The maximum frequency tolerance of the external clock input (SCLK) is 45% the frequency of RI2CCLK. RI2CCLK = 66.67 MHz, then set SCLK to 30 MHz or less.

The baud rate generator is a circuit that divides these clock signals according to the following formula.

$$\text{Baud Rate} = \frac{fc}{\text{Prescalar} \times \text{Divisor} \times 16}$$

- fc: Clock frequency of RI2CCLK or an external clock input (SCLK)
- Prescalar Value: 2, 8, 32, 128
- Divide Value: 1, 2, 3,…255

Table 9.1.2 shows example settings of divide values relative to representative baud rates.

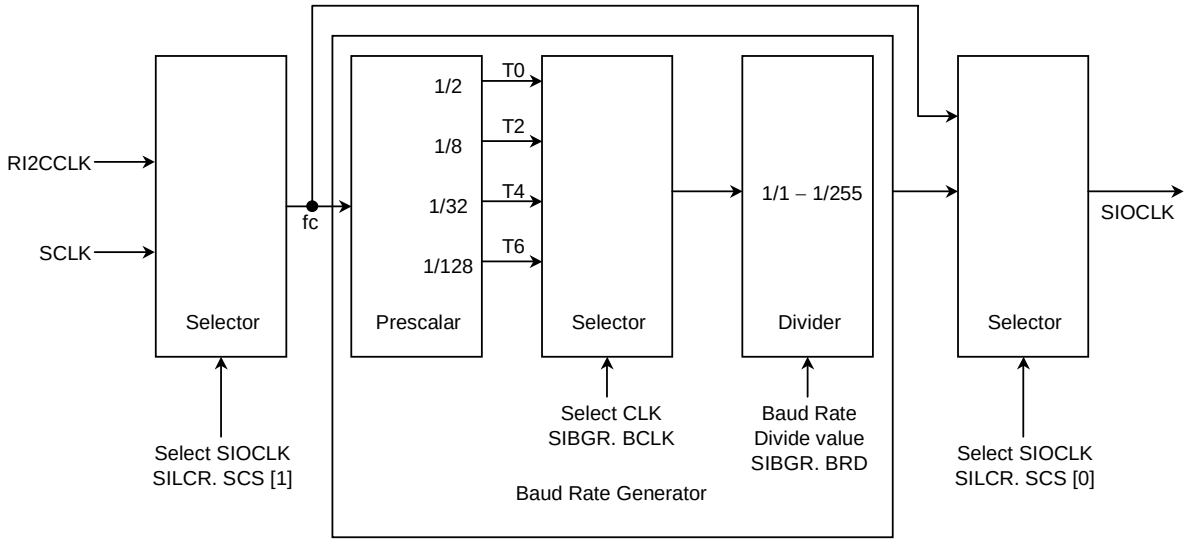


Figure 9.1.15 Baud Rate Generator and SIOCLK Generator

It is possible to correctly receive data if the error of the baud rate set by this controller is within 3.12 % of the target baud rate (communication baud rate).

Table 9.1.2 Example Divide Value Settings (and error [%] from target baud rate value)

fc [MHz]		Kbps	Prescalar Value (SIBGR.BLCK) and Divide Value (SIBGR.BRD)			
			2	8	32	128
RI2CCLK	66.6667	0.11				
		0.15				217 (0.01 %)
		0.30				109 (–0.45 %)
		0.60			217 (0.01 %)	54 (0.47 %)
		1.20			109 (–0.45 %)	27(0.47 %)
		2.40		217 (0.01 %)	54 (0.47 %)	14(–3.12%)
		4.80		109 (–0.45 %)	27(0.47 %)	7(–3.12%)
		9.60	217 (0.01 %)	54 (0.47 %)	14(–3.12%)	
		14.40	145 (–0.22 %)	36(0.47 %)	9(0.47%)	
		19.20	109 (–0.45 %)	27(0.47 %)	7(–3.12%)	
		28.80	72 (0.47 %)	18(0.47%)		
		38.40	54 (0.47 %)	14(–3.12%)		
		57.60	36(0.47 %)	9(0.47%)		
		76.80	27(0.47 %)	7(–3.12%)		
		115.20	18(0.47%)			
		230.40	9(0.47%)			
SCLK	7.3728	0.11			131 (–0.07 %)	33 (–0.83 %)
		0.15			96 (0.00 %)	24 (0.00 %)
		0.30			48 (0.00 %)	12 (0.00 %)
		0.60		96 (0.00 %)	24 (0.00 %)	6 (0.00 %)
		1.20		48 (0.00 %)	12 (0.00 %)	3 (0.00 %)
		2.40	96 (0.00 %)	24 (0.00 %)	6 (0.00 %)	
		4.80	48 (0.00 %)	12 (0.00 %)	3 (0.00 %)	
		9.60	24 (0.00 %)	6 (0.00 %)		
		14.40	16 (0.00 %)	4 (0.00 %)	1 (0.00 %)	
		19.20	12 (0.00 %)	3 (0.00 %)		
		28.80	8 (0.00 %)	2 (0.00 %)		
		38.40	6 (0.00 %)			
		57.60	4 (0.00 %)	1 (0.00 %)		
		76.80	3 (0.00 %)			
		115.20	2 (0.00 %)			

9.1.2.6 Data Reception

When the Serial Data Reception Disable bit (RSDE) of the Flow Control Register (SIFLCR) is set to "0", reception operation starts after the RXD signal start bit is detected. Start bits are detected when the RXD signal transitions from the High state to the Low state. Therefore, the RXD signal is not interpreted as a start bit if it is Low when the Serial Data Reception Disable bit is set to "0".

The received data are stored in the Receive FIFO. The Reception Data Full bit (RDIS) of the Interrupt Status Register (SIDISR) is set if the byte count of the stored reception data exceeds the value set by the Receive FIFO Request Trigger Level field (RDIL) of the FIFO Control Register (SIFCR).

An interrupt is signaled when the Reception Data Interrupt Enable bit (RIE) of the Interrupt Control Register (SIDICR) is set. The received data can be read from the Receive FIFO Data Register (SIRFIFO).

9.1.2.7 Data Transmission

Data stored in the Transmission Data FIFO are transmitted when the Serial Data Transmission Disable bit (TSDE) of the Flow Control Register (SIFLCR) is set to "0".

If the available space in the Transmit FIFO is greater than the byte count set by the Transmit FIFO Request Trigger Level (TDIL) of the Control Register (SIFCR), the transmission data empty bit (TDIS) of the Interrupt Status Register (SIDISR) is set.

An interrupt is signaled when the Transmission Data Interrupt Enable bit (TIE) of the Interrupt Control Register (SIDICR) is set.

9.1.2.8 Flow Control

SIO supports hardware flow control that uses the RTS*/CTS* signal.

The CTS* (Clear to Send) input signal indicates that data can be received from the reception side when it is Low. Setting the Transmission Enable Select bit (TES) of the Flow Control Register (SIFLCR) makes transmission flow control that uses the CTS* signal more effective.

It is also possible to generate status change interrupts by changing the state of the CTS* signal. The conditions in which interrupts are generated can be selected by the CTSS Active Condition field of the Interrupt Control Register (SIDICR).

Setting the RTS* (Request to Send) output signal to High requests the transmission side to pause transmission. Transmission resumes when the reception side becomes ready and the RTS* signal is set to Low.

Setting the Reception Enable Select bit (RCS) of the flow Control Register (SIFLCR) makes reception flow control that uses the RTS* signal more effective. The RTS* signal pin status becomes High when data of the byte count set by the RTS Active Trigger Level field (RTSTL) of the Flow Control Register (SIFLCR) accumulates in the Receive FIFO. The RTS* signal can also be made High by setting the RTS Software Control bit (RTSSC) of the Flow Control Register (SIFLCR). Setting this bit requests the transmission side to pause transmission.

9.1.2.9 Reception Data Status

Status data such as the following is also stored in the Receive FIFO.

- Overrun error

  An overrun error is generated if all 16-stage Receive FIFO buffers become full and more data is transferred to the Reception Read buffer. When this occurs, the Overrun Status bit is set by the last stage of the Receive FIFO.

- Parity error

  A parity error is generated when a parity error is detected in the reception data.

- Framing error

  A framing error is generated when "0" is detected at the first stop bit of the reception data.

- Break reception

  A break is detected when a framing error occurs in the reception data and all data in a single frame are "0". When this occurs, 2 frames (2 Bytes) of 0x00 data are stored in the Receive FIFO.

The Reception Error Interrupt bit (SIDISR.ERI) of the Interrupt Status Register (SIDISR) is set when one of the following errors is detected: an overrun error, a parity error, or a framing error. An interrupt is signaled if the Reception Error Interrupt Enable bit of the Interrupt Control Register (SIDICR) is set.

The Receive Break bit (RBRKD) and the Receiving Break bit (RBRKD) of the Status Change Interrupt Status Register (SISCISR) is set when a break is detected. The Receive Break bit (RBRKD) remains set until it is cleared by the software. The Receiving Break bit (RBRKD) is automatically cleared when a frame is received that is not a break.

The status of the next reception data to be read is set to the Overrun Error bit (UOER), Parity Error bit (UPER), Framing Error bit (UFER), and the Receive Break bit (RBRKD). Each of these statuses is updated when reception data is read from the Receive FIFO Register (SIRFIFO).

9.1.2.10 Reception Time Out

A Reception time out is detected and the Reception Time Out bit (TOUT) of the Interrupt Status Register (SIDISR) is set under the following conditions.

When at least 1 Byte of reception data exists in the Receive FIFO and the data reception time for the 2 frames (2 Bytes) after the last reception has elapsed.

9.1.2.11 Software Reset

It is necessary to reset the FIFO and perform a software reset in the following situations.

(1) After transmission data is set in FIFO, etc., transmission started but stopped before its completion

(2) An overrun occurred during data reception

Software reset is performed by setting the Software Reset bit (SWRST) of the FIFO Control Register (SIFCR). This bit automatically returns to "0" after initialization is complete. This bit must be set again since all SIO registers are initialized by software resets.

9.1.2.12 Error Detection/Interrupt Signaling

An interrupt is signaled if an error or an interrupt cause is detected, the corresponding status bit is set and the corresponding Interrupt Enable bit is set.

Figure 9.1.16 shows the relationship between the status bit for each interrupt cause and each interrupt enable bit. Please refer to the explanation for each status bit for more information about each interrupt cause.

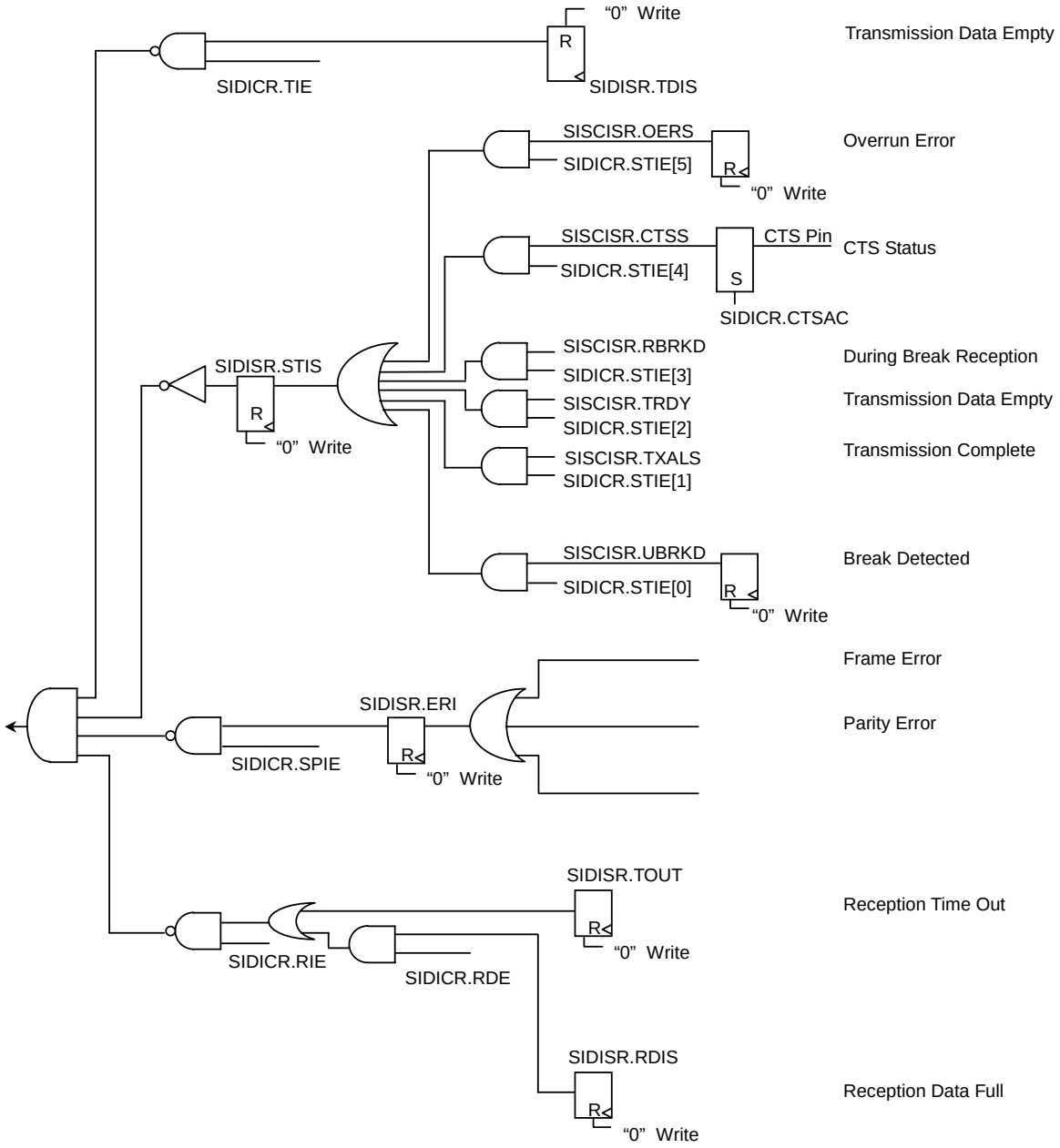


Figure 9.1.16 Relationship Between Interrupt Status Bits and Interrupt Signals

9.1.2.13 Multi-Controller System

The Multi-Controller System consists of one Master Controller, and multiple Slave Controllers as shown below in Figure 9.1.17.

In the case of the Multi-Controller System, the Master Controller transmits an address (ID) frame to all Slave Controllers, then transmits and receives data with the selected Slave Controller. Slave Controllers that were not selected will ignore this data.

Data frames whose data frame Wake Up bits (WUB) are "1" are handled as address (ID) frames. Data frames whose Wake Up bit (WUB) is "0" are handled as data frames.

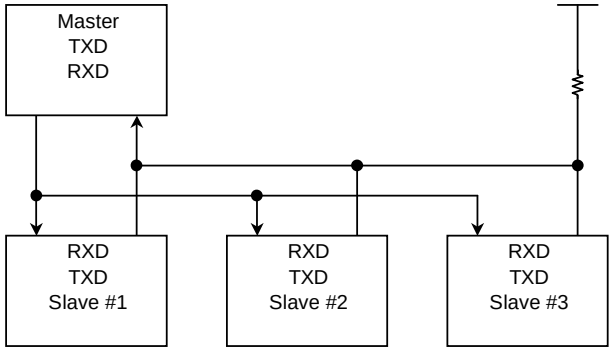


Figure 9.1.17 Example Configuration of Multi-Controller System

The data transfer procedure for the Multi-Controller System is as follows.

(1) The Master and Slave Controllers set the Mode field (UMODE) of the Line Control Register (SILCR) to "10" or "11" to set the Multi-Controller System mode. Also, the Slave Controller sets the open drain enable bit (UODE) of the Line Control Register (SILCR), setting the TXD output signal to open drain output.

(2) The Slave Controller sets the Reception Wake Up bit (RWUB) of the Line Control Register (SILCR), making it possible to receive address (ID) frames from the Master Controller.

(3) The Master Controller sets the Transmission Wake Up bit (TWUB) of the Line Control Register (SILCR), and transmits the address (ID) of the selected Slave Controller. This causes the address (ID) frame to be transmitted. The Reception after Address Transmission Wake Up bit (RWUB) is cleared, enabling reception of data frames.

(4) Since the Reception Wake Up bit (RWUB) is set, the Slave Controller generates an interrupt to the CPU by receiving an address (ID) frame. The CPU compares its own address (ID) and the received data together. If they match, the Reception Wake Up bit (RWUB) is cleared, making data frame reception possible.

(5) The Master Controller and the selected Slave Controller clear the Transmission Wake Up bit (TWUB) of the Line Control Register (SILCR), then set the mode that transmits data frames.

(6) Transmit/Receive data between the Master Controller and the selected Slave Controller. Then, Slave Controllers that were not selected ignore data frames since the Reception Wake Up bit (RWUB) is still set.

9.1.3 GPIO Port Functions

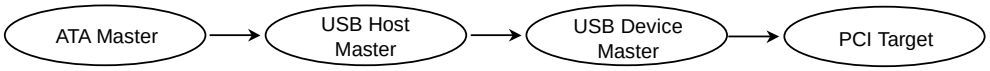
- Up to 5 GPIO pins available
- Independent selection of direction of each GPIO pin
- Independent choice of totem-pole or open-drain for all GPIO outputs
- ALL 5 GPIO pins can be read regardless of direction or mode

9.1.4 Internal G-bus Arbiter Functions

Fixed priority (FRS = 1)

As shown below, ATA Master has the highest priority and USB Device Master has the lowest priority when Fixed Priority is selected and G-bus Arbiter setting Register D[5:1]is 0x00.

ATA > USB host > USB device>PCI Target



a) Example when Fixed Priority is selected and G-bus Arbiter setting Register D[5:1]is 0x00.

Round Robin method (FRS = 0)

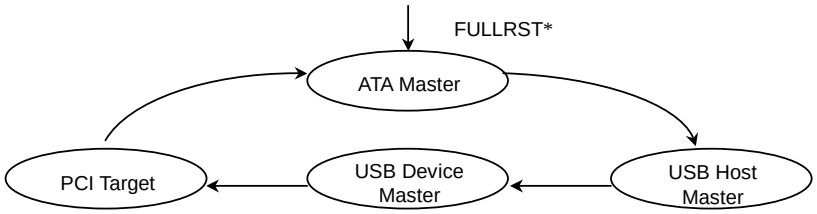
The last device to perform DMA transfer has the lowest priority.

After ATA DMA transfer execution: USB Host > USB Device > PCI Target > ATA

After USB Host DMA transfer execution: USB Device > PCI Target > ATA > USB Host

After USB Device DMA transfer execution: PCI Target > ATA > USB Host > USB Device

After PCI Target transfer execution: ATA > USB Host > USB Device > PCI Target



b) Round Robin Priority is selected

As shown in the following figure, the internal PCI arbiter has a function for rotating the priority. The last agent granted is always dropped to the bottom of the queue for the next arbitration cycle.

For example, if USB Host and USB device bus master requests conflict when the USB Host Master has priority, the USB Host Master request is given priority then the USB Device Master becomes the next priority function when GNT* is asserted during this contention.

9.2 Register Description

Table 9.2.1 Register Access Type

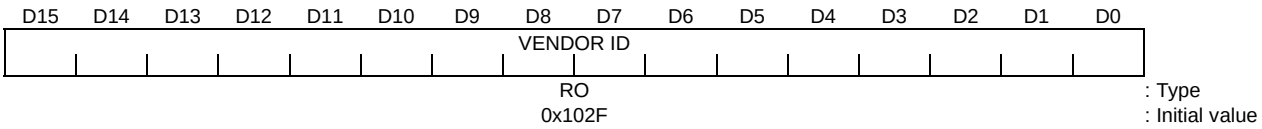
Access Type	Meaning
RO	Read-only
WO	Write-only
RW	Can both be read and written to.
RWC	Can both be read and written to. Cleared when written to.

9.2.1 Configuration Map

23	15	07	00	
Device ID		Vendor ID		0x00
Status		Command		0x04
Class Code			Revision ID	0x08
BIST	Header Type	Latency Timer	Cache Line Size	0x0C
I2C I/O base register				0x10
SIO I/O base register				0x14
GPIO I/O base register				0x18
Internal G-bus Arbiter I/O base register				0x1C
				0x20
				0x24
				0x28
Sub System ID		Sub Vendor ID		0x2C
				0x30
			Capability Pointer	0x34
				0x38
Maximum Latency	Minimum GNT	PCI Interrupt Pin	PCI Interrupt Line	0x3C
		Retry Timeout	TRDY Timeout	0x40
			PCI Interrupt Pin Select	0x44
				0x48-0xD8
Power Management Capabilities		Next Item Pointer	Capability ID	0xDC
Data	PMCSR BSE	Power Management Control/Status		0xE0
				0xE4-0xFC

9.2.2 Configuration Register Details

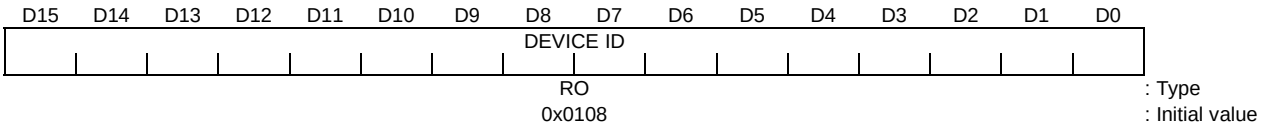
9.2.2.1 Vendor ID Register (0x00–0x01)



Bits	Mnemonic	Field Name	Description
D15:D0	VENDOR ID	Vendor ID	Vendor ID This value 0x102F represents the Toshiba Semiconductor Company.

Figure 9.2.1 Vendor ID Register

9.2.2.2 Device ID Register (0x02–0x03)



Bits	Mnemonic	Field Name	Description
D15:D0	DEVICE ID	Device ID	Device ID This value 0x0108 represents the I²C Bus/SIO/GPIO Controller.

Figure 9.2.2 Device ID Register

9.2.2.3 Command Register (0x04–0x05)

D15	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	
Reserved		FBBE	SER	WCC	PER	VGA	MWI	SP	BM	MS	IOS	
		RW	RW	RO	RW	RO	RW	RO	RO	RO	RW	: Type
		0	0	0	0	0	0	0	0	0	1	: Initial value

Bits	Mnemonic	Field Name	Description
D15:D10		Reserved	
D9	FBBE	Fast Back-to-Back Enable	Fast Back-to-Back Enable This bit controls whether or not a master can do fast back-to-back transactions to different devices. A value of 1 means the master is allowed to generate fast back-to-back transactions to different agents. A value of 0 means fast back-to-back transactions are only allowed to the same agent.
D8	SER	SERR Enable	SERR Enable This is the enable bit for the SERR* driver. All devices that have an SERR* pin must implement this bit, also bit[6] must be on to report address parity errors when SERR* is "1". 1: Device must take its normal action when a parity error id detected. 0: Device must set its parity errors detection status bit (bit[15] in the status register) when an error occurred but does not assert PERR* and continues its normal operation (Default).
D7	WCC	Wait Cycle Control	Wait Cycle Control Fixed to "0" since the address/data are not stepped.
D6	PER	Parity Error Response	Parity Error Response 1: Enable SERR* driver 0: Disable SERR* driver (Default)
D5	VGA	VGA Palette Snoop	VGA Palette Snoop Fixed to "0" since this module is not a VGA device.
D4	MWI	Memory Write and Invalidate Enable	Memory Write and Invalidate Enable This module does not support memory write-and-invalidate commands. Writing this bit has no effect.
D3	SP	Special Cycle	Special Cycle Fixed to "0" since this module does not respond to special cycles.
D2	BM	Bus Master	Bus Master Set this bit to "0" to not operate this module as a PCI bus master.
D1	MS	Memory Space	Memory Space Fixed to "0" since this module does not have memory space.
D0	IOS	I/O Space	I/O Space Set this bit to "1" to support access to I2C, SIO and GPIO I/O registers.

Figure 9.2.3 Command Register

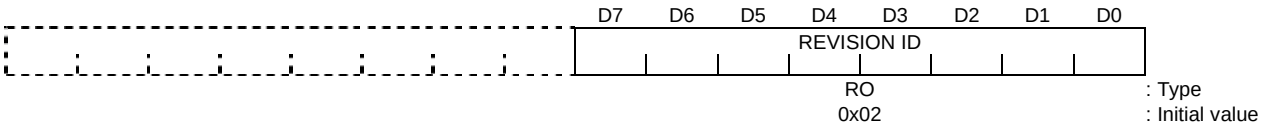
9.2.2.4 Status Register (0x06–0x07)

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3				D0	
DPE	SSE	RMA	RTA	STA	DECT		DPD	FBBC	Reserved		CAP	Reserved					
RWC	RWC	RO	RO	RWC	RO		RO	RO	RO		RO	RO					: Type
0	0	0	0	0	01		0	1	0	0	1	0	0	0	0		: Initial value

Bits	Mnemonic	Field Name	Description
D15	DPE	Detected Parity Error	Detected Parity Error (RWC) This bit must be set by the device whenever it detects a parity error, even if parity error handling is disabled (as controlled by bit[6] in the Command register bit). 1: Reset 0: No action (Default)
D14	SSE	Signaled System Error	Signaled System Error (RWC) This bit must be set whenever the device asserts SERR*. Devices who will never assert SERR* do not need to implement this bit. 1: SERR* asserted 0: No SERR* asserted (Default)
D13	RMA	Received Master Abort	Received Master Abort Fixed to “0” since this bit is only implemented by Bus master.
D12	RTA	Received Target Abort	Received Target Abort Fixed to “0” since this bit is only implemented by Bus master.
D11	STA	Signaled Target Abort	Signaled Target Abort If target abort occurs, this bit is set to 1 when this module is a PCI target.
D10:D9	DECT	DEVSEL Timing	DEVSEL Timing Specifies DEVSEL response timing. These bits are fixed to “01b” since this module responds using the medium DEVSEL timing.
D8	DPD	Master Data Parity Detected	Master Data Parity Detected Fixed to “0” since this bit is only Implemented by Bus master.
D7	FBBC	Fast Back-to-Back Capable	Fast Back-to-Back Capable This bit indicates whether or not target is capable of accepting fast back-to-back transactions when the transactions are not to the same agent. This bit is fixed to “1” since the device can accept these transactions.
D6:D5		Reserved	
D4	CAP	Capabilities	Capabilities Set this bit to “1” to support PCI Power Management. This function is supported by default.
D3:D0		Reserved	

Figure 9.2.4 Status Register

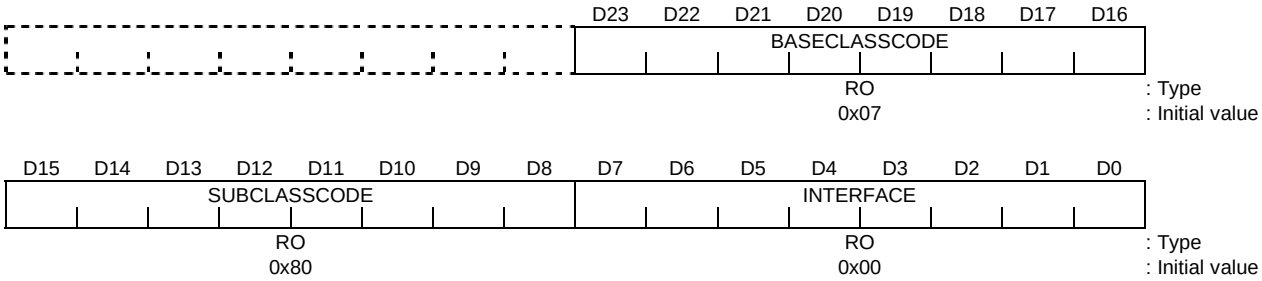
9.2.2.5 Revision ID Register (0x08)



Bits	Mnemonic	Field Name	Description
D7:D0	Revision ID	Revision ID	Revision ID This value 0x02 represents the revision.

Figure 9.2.5 Revision ID Register

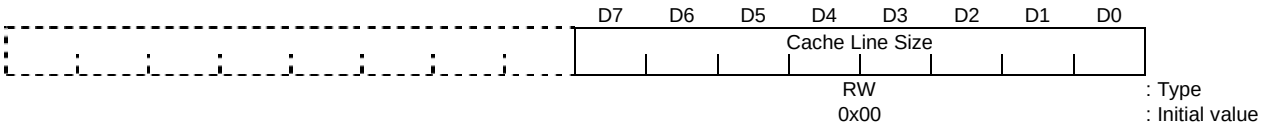
9.2.2.6 Class Code Register (0x09–0x0B)



Bits	Mnemonic	Field Name	Description
D23:D16	BASECLASS CODE	Base Class Code	Class Code This value 0x078000 represents other communications device (I²C, SIO, GPIO).
D15:D8	SUBCLASSC ODE	Sub Class Code	
D7:D0	INTERFACE	Interface	

Figure 9.2.6 Class Code Register

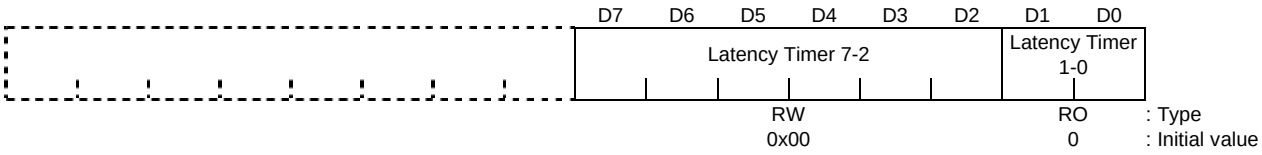
9.2.2.7 Cache Line Size Register (0x0C)



Bits	Mnemonic	Field Name	Description
D7:D0	Cache Line Size	Cache Line Size	Cache Line Size This register specifies the system cache line size in units of DWORDs. Writing this register has no effect.

Figure 9.2.7 Cache Line Size Register

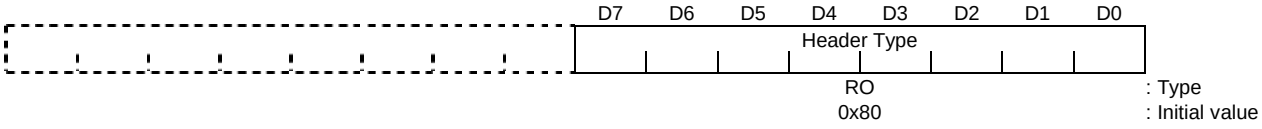
9.2.2.8 Latency Timer Register (0x0D)



Bits	Mnemonic	Field Name	Description
D7:D0	Latency Timer	Latency Timer	Latency Timer This register specifies, in units of PCI bus clocks, the value of the Latency Timer for this PCI bus master. Writing this register has no effect.

Figure 9.2.8 Latency Timer Register

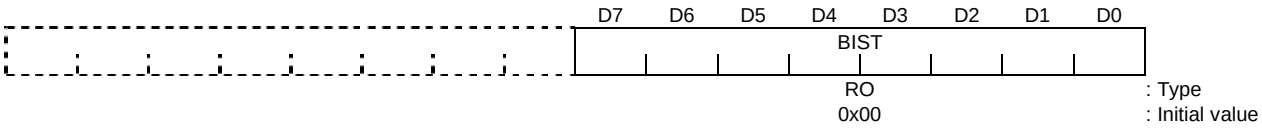
9.2.2.9 Header Type Register (0x0E)



Bits	Mnemonic	Field Name	Description
D7:D0	Header Type	Header Type	Header Type This value 0x80 represents a Multifunction device.

Figure 9.2.9 Header Type Register

9.2.2.10 Built-in Self Test Register (0x0F)



Bits	Mnemonic	Field Name	Description
D7:D0	BIST	Built-in Self Test	Built-in Self Test Fixed to "0" since BIST is not supported.

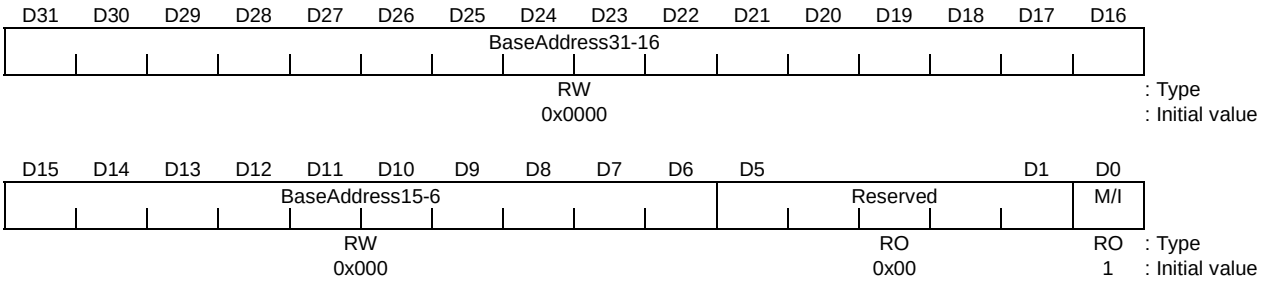
Figure 9.2.10 Built-in Self Test Register

9.2.2.11 I²C IO Base Address Register (0x10-0x13)

Bits	Mnemonic	Field Name	Description
D31:D4	BaseAddress	Base Address	Base Address This field is the upper 28bits of the I²C IO Register base address.
D3:D1		Reserved	Fixed to "000" since the IO address is opened in 16-byte units.
D0	M/I	IO space Indicator	IO space Indicator Fixed to "1" since the base address register is used for I/O.

Figure 9.2.11 I²C IO Base Address Register

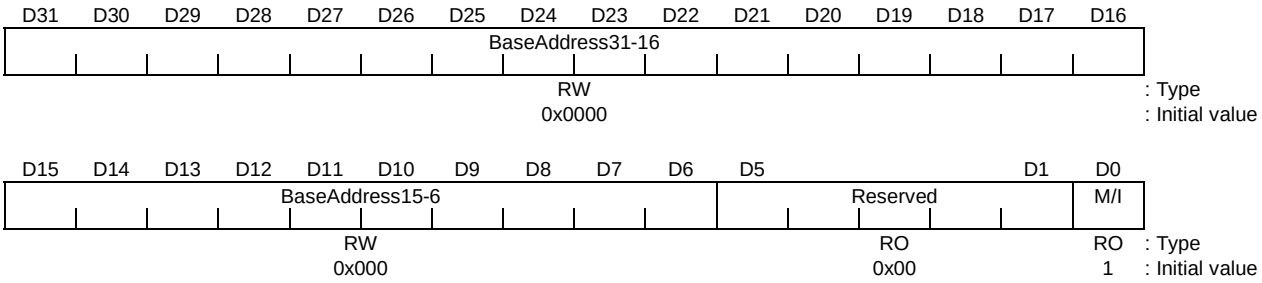
9.2.2.12 SIO IO Base Address Register (0x14-0x17)



Bits	Mnemonic	Field Name	Description
D31:D6	BaseAddress	Base Address	Base Address This field is the upper 26bits of the SIO IO Register base address.
D5:D1		Reserved	Reserved Fixed to “0” since the IO address is opened in 64-byte units.
D0	M/I	IO space Indicator	IO space Indicator Fixed to “1” since the base address register is used for I/O.

Figure 9.2.12 SIO IO Base Address Register

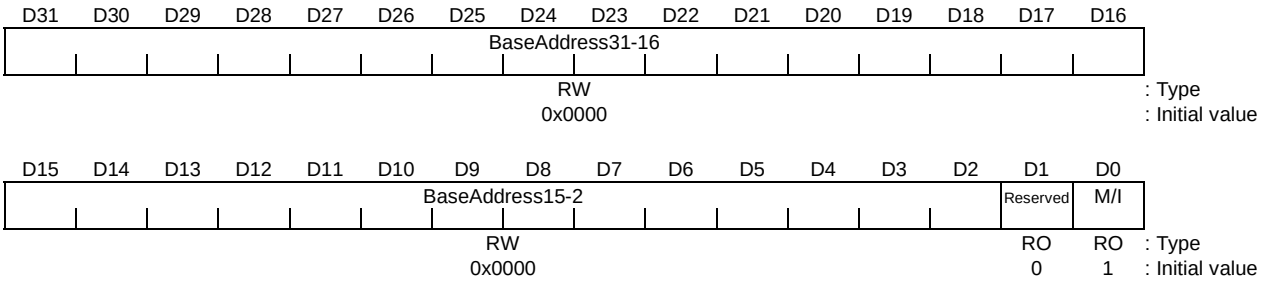
9.2.2.13 GPIO IO Base Address Register (0x18-0x1B)



Bits	Mnemonic	Field Name	Description
D31:D6	BaseAddress	Base Address	Base Address This field is the upper 26bits of the GPIO IO Register base address.
D5:D1		Reserved	Fixed to “0” since the IO address is opened in 64-byte units.
D0	M/I	IO space Indicator	IO space Indicator Fixed to “1” since the base address register is used for I/O.

Figure 9.2.13 GPIO IO Base Register

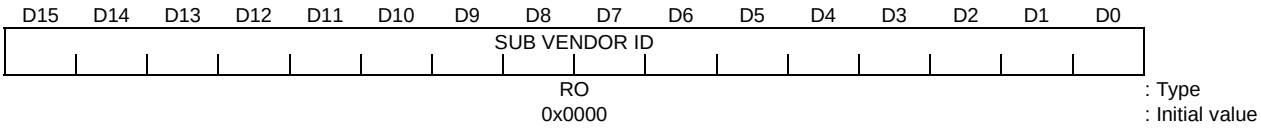
9.2.2.14 Internal G-bus Arbiter IO Base Address Register (0x1C-0x1F)



Bits	Mnemonic	Field Name	Description
D15:D2	BaseAddress	Base Address	Base Address This field is the upper 30bits of the Internal G-bus Arbiter IO Register base address.
D1		Reserved	Reserved Fixed to"0" since the IO address is opened in 4-byte units.
D0	M/I	IO space Indicator	IO space Indicator Fixed to "1" since the base address register is used for I/O.

Figure 9.2.14 Internal G-bus Arbiter IO Base Address Register

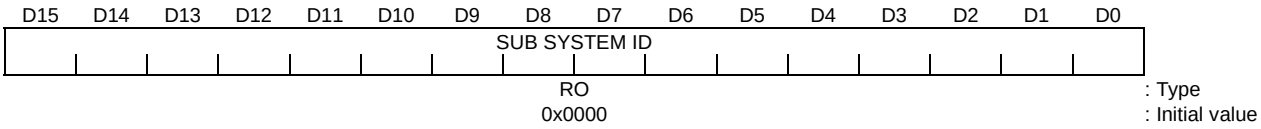
9.2.2.15 Sub Vendor ID Register (0x2C–0x2D)



Bits	Mnemonic	Field Name	Description
D15:D0	SUB VENDOR ID	Sub Vendor ID	Sub Vendor ID If use Power On Loading function, please refer to 10. Loadable PCI Configuration Space.

Figure 9.2.15 Sub Vendor ID Register

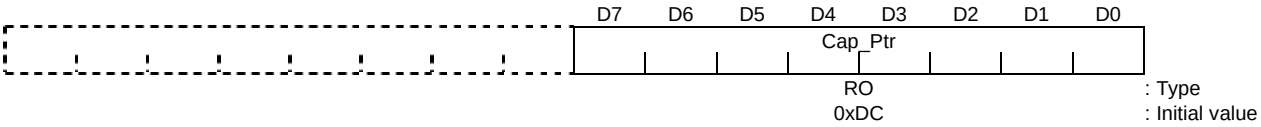
9.2.2.16 Sub System ID Register (0x2E–0x2F)



Bits	Mnemonic	Field Name	Description
D15:D0	SUB SYSTEM ID	Sub System ID	Sub System ID If use Power On Loading function, please refer to 10. Loadable PCI Configuration Space.

Figure 9.2.16 Sub System ID Register

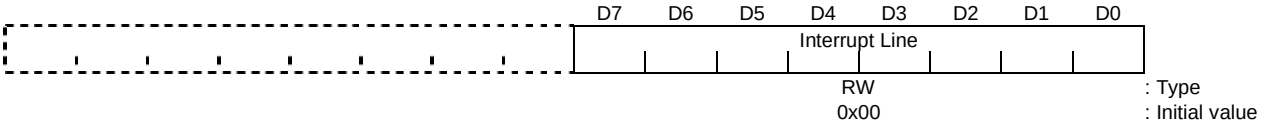
9.2.2.17 Capability Pointer Register (0x34)



Bits	Mnemonic	Field Name	Description
D7:D0	Cap_Ptr	Capability Pointer	Capability Pointer This is the address of the PCI power management register block.

Figure 9.2.17 Capability Pointer Register

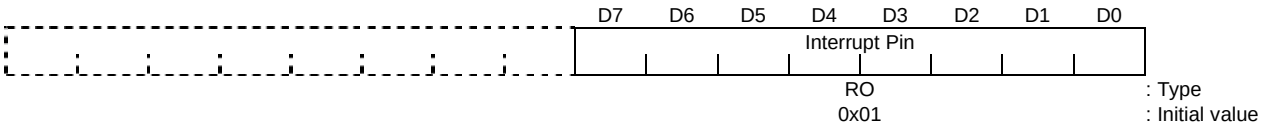
9.2.2.18 PCI Interrupt Line Register (0x3C)



Bits	Mnemonic	Field Name	Description
D7:D0	Interrupt Line	Interrupt Line	Interrupt Line These bits specify an IRQ interrupt number.

Figure 9.2.18 PCI Interrupt Line Register

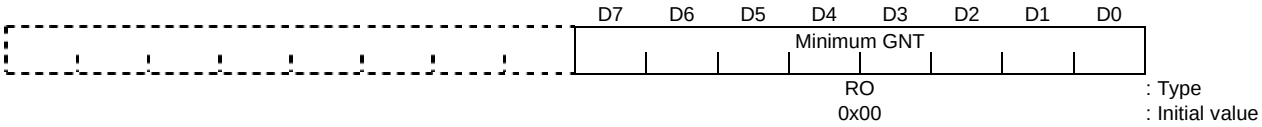
9.2.2.19 PCI Interrupt Pin Register (0x3D)



Bits	Mnemonic	Field Name	Description
D7:D0	Interrupt Pin	Interrupt Pin	Interrupt Pin This register tells which Interrupt PIN the device uses. It can be specified by the PCI Interrupt PIN Selectable Register (0x44).

Figure 9.2.19 PCI Interrupt Pin Register

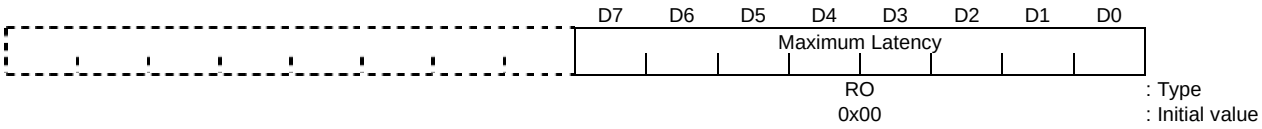
9.2.2.20 Minimum GNT Register (0x3E)



Bits	Mnemonic	Field Name	Description
D7:D0	Minimum GNT	Minimum GNT	Minimum GNT This is the minimum burst time that this module requires. Specify it in units of 0.25 µs. The minimum burst time for this module is fixed to “0x00.”

Figure 9.2.20 Minimum GNT Register

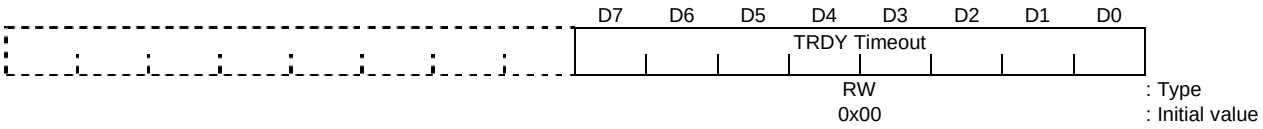
9.2.2.21 Maximum Latency Register (0x3F)



Bits	Mnemonic	Field Name	Description
D7:D0	Maximum Latency	Maximum Latency	Maximum Latency This is the maximum wait time during PCI access. Specify it in units of 0.25 µs. The maximum wait time for this module is fixed to 0x00.

Figure 9.2.21 Maximum Latency Register

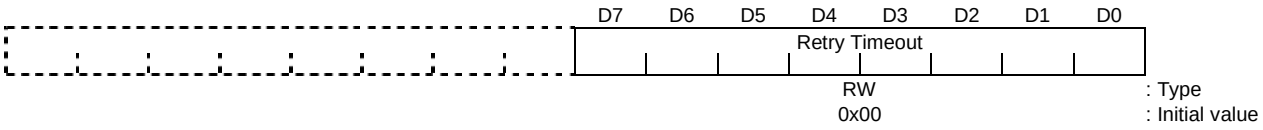
9.2.2.22 TRDY Timeout Register (0x40)



Bits	Mnemonic	Field Name	Description
D7:D0	TRDY Time out	TRDY Time out	TRDY Time out Value Sets number of PCI clocks that core as Master will wait for TRDY. Writing this register has no effect.

Figure 9.2.22 TRDY Timeout Register

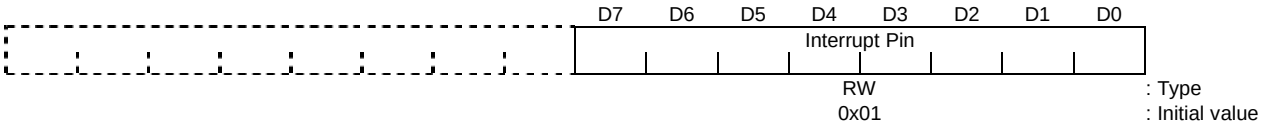
9.2.2.23 Retry Timeout Register (0x41)



Bits	Mnemonic	Field Name	Description
D7:D0	Retry Time out	Retry Time out	Retry Timeout Value Sets number of retries that the core as Master will perform. Writing this register has no effect.

Figure 9.2.23 Retry Timeout Register

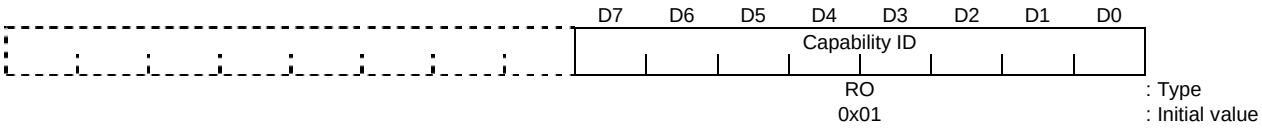
9.2.2.24 PCI Interrupt Pin Selectable Register (0x44)



Bits	Mnemonic	Field Name	Description
D7:D0	Interrupt Pin	Interrupt Pin	Interrupt Pin These bits set a PCI interrupt. 01: INTA* 02: INTB* Others: Default (INTA*)

Figure 9.2.24 PCI Interrupt Pin Selectable Register

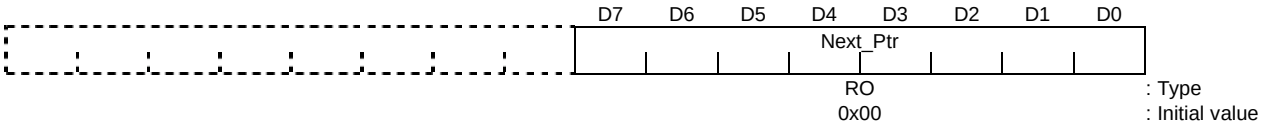
9.2.2.25 Capability ID Register (0xDC)



Bits	Mnemonic	Field Name	Description
D7:D0	Capability ID	Capability ID	Capability ID Fixed to “1.” Represents the PCI power management register block.

Figure 9.2.25 Capability ID Register

9.2.2.26 Next Item Pointer Register (0xDD)



Bits	Mnemonic	Field Name	Description
D7:D0	Next_Ptr	Next Item Pointer	Next Item Pointer Fixed to “0x00” since no next block exists.

Figure 9.2.26 Next Item Pointer Register

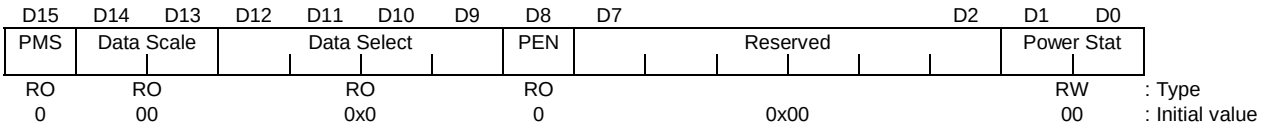
9.2.2.27 Power Management Capabilities PMC (0xDE–0xDF)

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	
PMES					D2S	D1S	AuxC	Aux Current		DSI	Reserved	PMC	VER			
RO					RO	RO	RO	RO		RO		RO	RO			: Type
0x00					0	0	0	00		0	0	0	010			: Initial value

Bits	Mnemonic	Field Name	Description
D15:D11	PMES	PME Support	PME Support Set these bits to “0” since PME* is not supported.
D10	D2S	D2 Support	D2 Support Set this bit to “0” since D2 is not supported.
D9	D1S	D1 Support	D1 Support Set this bit to “0” since D1 is not supported.
D8:D6	Aux Current	Aux Current	Aux Current These bits set the auxiliary current value required for the 3.3 V auxiliary power supply. Fixed to “0” since PME* is not supported.
D5	DSI	Device Specific Initialization	Device Specific Initialization
D4		Reserved	
D3	PMC	PME Clock	PME Clock Clear this bit to “0” since PCI clock is not required to assert PME*.
D2:D0	VER	Version	Version These bits set a version number, which is “0x02” indicating that the module conforms to PCI Power Management Interface Specifications 1.1.

Figure 9.2.27 Power Management Capabilities PMC

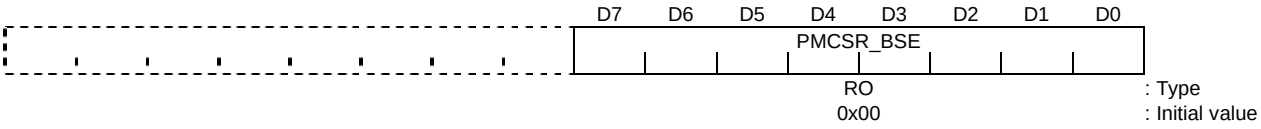
9.2.2.28 Power Management Control/Status Register PMCSR (0xE0–0xE1)



Bits	Mnemonic	Field Name	Description
D15	PMS	PME Status	PME Status Fixed to “0” since PME* is not supported.
D14:D13	Data Scale	Data Scale	Data Scale Fixed to “0x00” since Data Scale is not supported.
D12:D9	Data Select	Data Select	Data Select Fixed to “0x00” since Data Select is not supported.
D8	PEN	PME Enable	PME Enable Fixed to “0” since PME* is not supported.
D7:D2		Reserved	
D1:D0	Power Stat	Power Status	Power Status The OS uses these bits to set the current device status. 00: D0 11: D3

Figure 9.2.28 Power Management Control/Status Register PMCSR

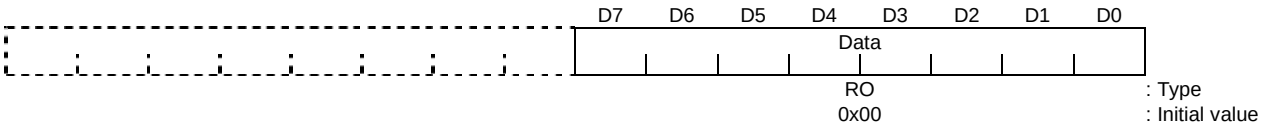
9.2.2.29 PCI to PCI Bridge Support Extension Register PMCSR BER (0xE2)



Bits	Mnemonic	Field Name	Description
D7:D0	PMCSR_BSE	MCSR_BSE	MCSR_BSE Set this bit to “0” since this module is not a PCI-PCI bridge.

Figure 9.2.29 PCI to PCI Bridge Support Extension Register PMCSR BER

9.2.2.30 Data (0xE3)



Bits	Mnemonic	Field Name	Description
D7:D0	Data	Data	Data Fixed to “0” since the Data Select is not supported.

Figure 9.2.30 Data

9.2.3 I²C I/O Registers

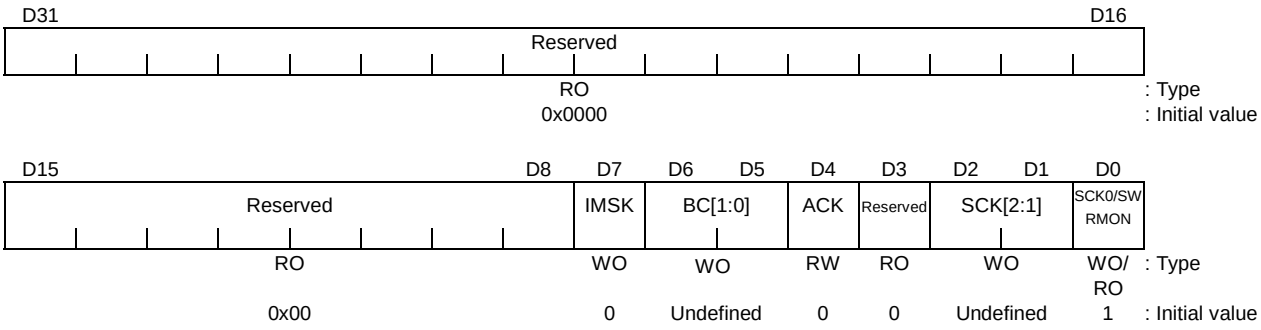
This register is accessed according to the offset from the base address specified by the I²C I/O Base Address Register of the PCI Configuration Register.

GPIO, SIO and I²C are Multiplexed each pins. Their functions can select by Mode Select Resister in the GPIO module.

Table 9.2.2 I²C I/O Register List

Offset Address	Size	Register	R/W
0x00	4	Control 1 (SBICR1)	RW
0x04	4	Control 2 (SBICR2)	RW
0x08	4	I²C Bus Address (I2CAR)	WO
0x0C	4	Data (SBIDBR)	RW

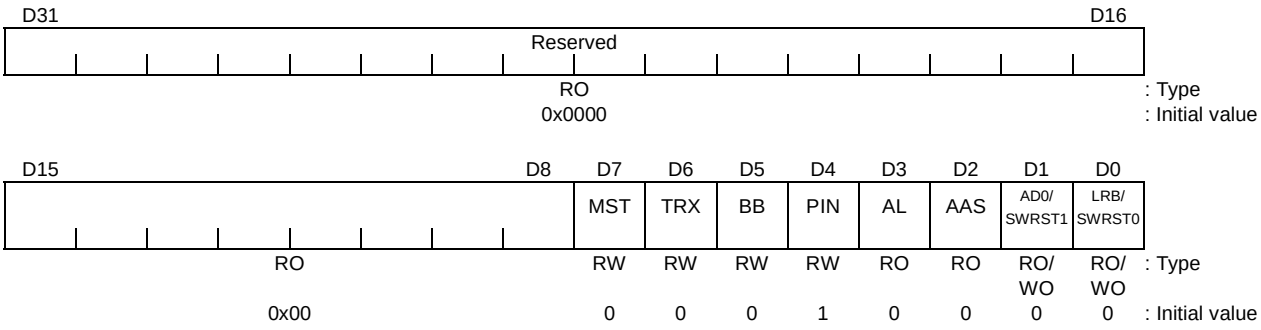
9.2.3.1 Control 1 Register (SBICR1, offset 0x00-0x03)



Bits	Mnemonic	Field Name	Description
D31:D8		Reserved	
D7	IMSK	IMSK	IMSK I²C Interrupt Mask 0: Does not mask I²C interrupt. 1: Mask I²C interrupt.
D6:D5	BC[1:0]	BC[1:0]	BC[1:0] Selects the transfer bit count. Set only 00 to BC[1:0] for the number of the next data item. BC[1:0] 00 01 ACK=0: Clock Count 8 1 Data Length 8 1 ACK=1: Clock Count 9 2 Data Length 8 1
D4	ACK	ACK	ACK ACK clock. 0: Does not generate the ACK clock. 1: Generates the ACK clock.
D3		Reserved	
D2:D0	SCK[2:0]	SCK[2:0]	SCK[2:0] Select the SCL Output clock frequency. $\text{SCL Output Clock} = \frac{Fc/4}{2^n+10}$ [Hz] (Fc = 66.67 MHz) SCL Output Clock (SCL kHz) SCK[2:0] 000 001 010 011 100 101 110 111 N 5 6 7 8 9 10 11 --- SCL Output Clock (kHz) 396.8 225.2 120.8 62.7 31.9 16.1 8.1 --- Note: Must not set 111 to SCK.
D0	SWRMON	SWRMON	SWRMON Software Reset Status Monitor. 0: Software reset is in progress 1: Software reset has been cleared

Figure 9.2.31 Control 1 Register

9.2.3.2 Control 2 Register (SBICR2, offset 0x04-0x07)



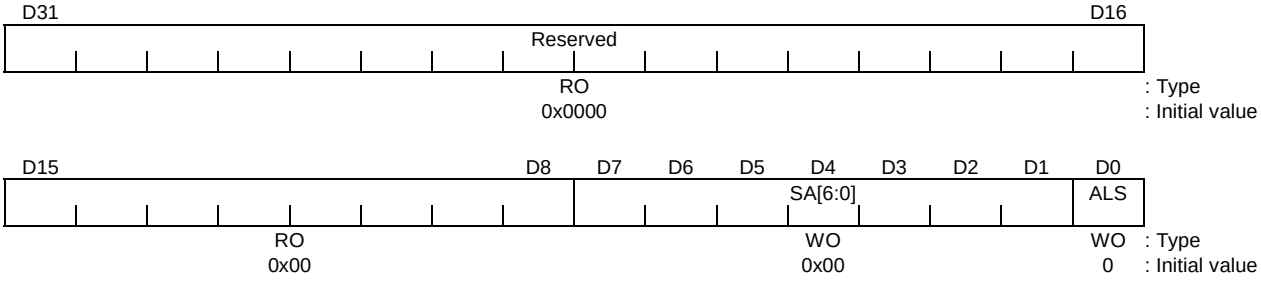
Bits	Mnemonic	Field Name	Description
D31:D8		Reserved	
D7	MST	MST	MST Master /Slave Select Monitor 0: Slave 1: Master
D6	TRX	TRX	TRX Transmit/Receive Select Monitor 0: Receive 1: Transmit
D5	BB	BB	BB I²C Bus Status Monitor Read 0 Bus is free 1 Bus is busy Write 0 Stop state occurred 1 Start state occurred
D4	PIN	PIN	PIN Interrupt Request Reading/Writing SBIDBR when in the Byte Transfer mode automatically clears interrupts. Read 0 Interrupt State Occurred state 1 Interrupt Request Cleared state Write 0 --- 1 Interrupt request was cleared
D3	AL	AL	AL Arbitration Lost Detect Fixed to “0” since Detect Arbitration Lost is not supported.
D2	AAS	AAS	AAS Slave Address Match Detect 0: --- 1: Slave address match or general call was detected.

Figure 9.2.32 Control 2 Register (1/2)

Bits	Mnemonic	Field Name	Description
D1	AD0	AD0	AD0 General Call Detect 0: --- 1: Detect general calls
D0	LRB	LRB	LRB Last Reception Bit Monitor
D1:D0	SWARST[1:0]	SWARST[1:0]	SWARST[1:0] Software Reset Occurred A reset occurs when “10” is first written and then “01” is written.

Figure 9.2.32 Control 2 Register (2/2)

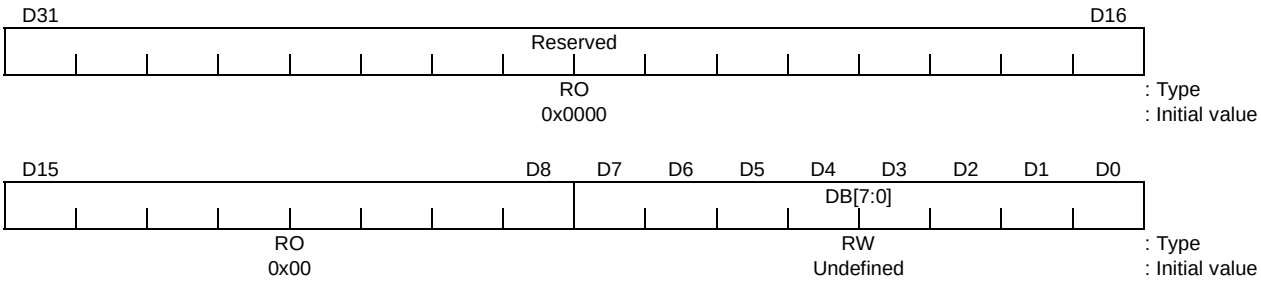
9.2.3.3 I2C Bus Address Register (I2CAR, offset 0x08-0x0B)



Bits	Mnemonic	Field Name	Description
D15:D8		Reserved	
D7:D1	SA[6:0]	SA[6:0]	SA[6:0] Set Slave address when operating as a slave device.
D0	ALS	ALS	ALS Set Address Acknowledge mode. 0: Acknowledge slave address 1: Do not acknowledge slave address

Figure 9.2.33 I2C Bus Address Register

9.2.3.4 Data Register (SBIDBR, offset 0x0C-0x0F)



Bits	Mnemonic	Field Name	Description
D31:D8		Reserved	
D7:D0	DB[7:0]	DB[7:0]	DB[7:0] This register is used to store transmission data writes and reception data. When in the I²C mode, pack data to the MSB (bit[7]) side and write when transmitting data in bit units.

Figure 9.2.34 Data Register

9.2.4 SIO I/O Registers

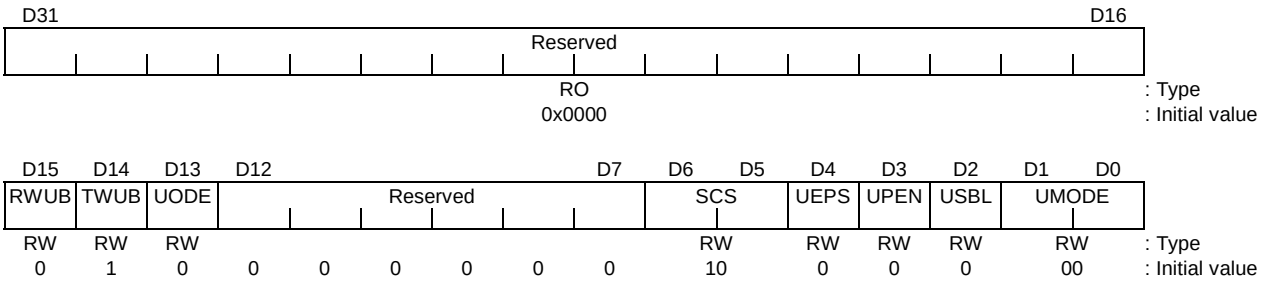
This register is accessed according to the offset from the base address specified by the SIO I/O Base Address Register of the PCI Configuration Register.

GPIO, SIO and I^2C are Multiplexed each pins. Their functions can select by Mode Select Resister in the GPIO module.

Table 9.2.3 SIO I/O Register List

Offset Address	Size	Register	R/W
0x00	4	Line Control Register	RW
0x04	4	Interrupt Control Register	RW
0x08	4	Interrupt Status Register	RW
0x0C	4	Status Change Interrupt Status Register	RW
0x10	4	FIFO Control Register	RW
0x14	4	Flow Control Register	RW
0x18	4	Baud Rate Control Register	RW
0x1C	4	Transmit FIFO Register	WO
0x20	4	Receive FIFO Register	RO
0x24-0x3F	0x1C	Reserved (fixed to “0”)	RO

9.2.4.1 Line Control Register (SILCR, offset 0x00-0x03)



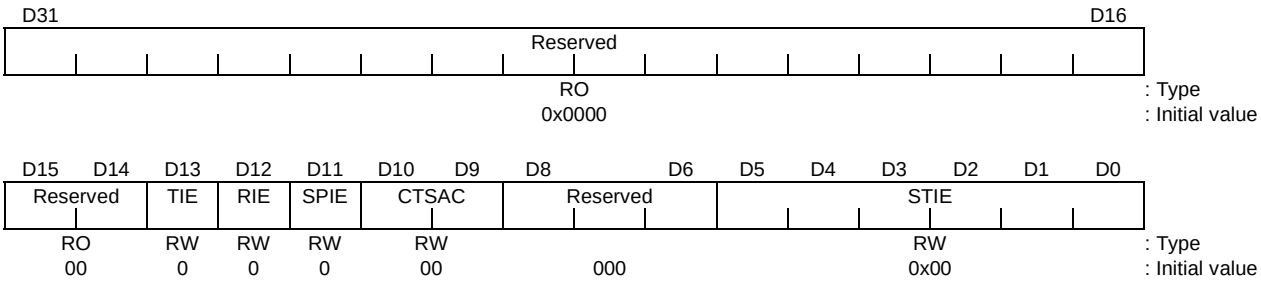
Bits	Mnemonic	Field Name	Description
D31:D16		Reserved	
D15	RWUB	Receive Wake Up Bit	Receive Wake Up Bit Wake Up Bit for Receive (Default: 0) When in the Multi-Controller System mode, this field selects whether to receive address (ID) frames whose Wake Up bits (WUB) are “1” or to receive data frames whose Wake Up bits (WUB) are “0”. This value is undefined when not in the Multi-Controller System mode. 0: Receive data frames 1: Receive address (ID) frames
D14	TWUB	Transmit Wake Up Bit	Transmit Wake Up Bit Wake Up Bit for Transmit (Default: 1) When in the Multi-Controller System mode, this field specifies the Wake Up bit (WUB). This value is undefined when not in the Multi-Controller System mode. 0: Data frame transfer (WUB = 0) 1: Address (ID) frame transfer (WUB = 1)
D13	UODE	Open Drain Enable	Open Drain Enable TXD Open Drain Enable (Default: 0) This field selects the output mode of the TXD signal. When in the Multi-Controller System mode, the Slave Controller must set the TXD signal to Open Drain. 0: Totem pole output 1: Open drain output
D12:D7		Reserved	
D6:D5	SCS	Clock Select	Clock Select SIO Clock Select (Default: 00) This field selects the serial transfer clock. The clock frequency that is the serial transfer clock divided by 16 becomes the baud rate (bps). 00: Internal clock (IMBUSCLK) 01: Baud rate generator output that divided IMBUSCLK 10: External clock (SCLK) 11: Baud rate generator output that divided SCLK

Figure 9.2.35 Line Control Register (1/2)

Bits	Mnemonic	Field Name	Description
D4	UEPS	Even Parity Select	Even Parity Select UART Even Parity Select (Default: 0) This field selects the parity mode. 0: Odd parity 1: Even parity
D3	UPEN	Parity Check Enable	Parity Check Enable UART Parity Enable (Default: 0) This field selects whether to perform the parity check. 0: Disable the parity check 1: Enable the parity check
D2	USBL	Stop Bit Length	Stop Bit Length UART Stop Bit Length (Default: 0) This field specifies the stop bit length. 0: 1 bit 1: 2 bits
D1:D0	UMODE	Mode	Mode UART Mode (Default: 00) This field sets the data frame mode. 00: 8-bit data length 01: 7-bit data length 10: Multi-Controller 8-bit data length 11: Multi-Controller 7-bit data length

Figure 9.2.35 Line Control Register (2/2)

9.2.4.2 Interrupt Control Register (SIDICR, offset 0x04-0x07)



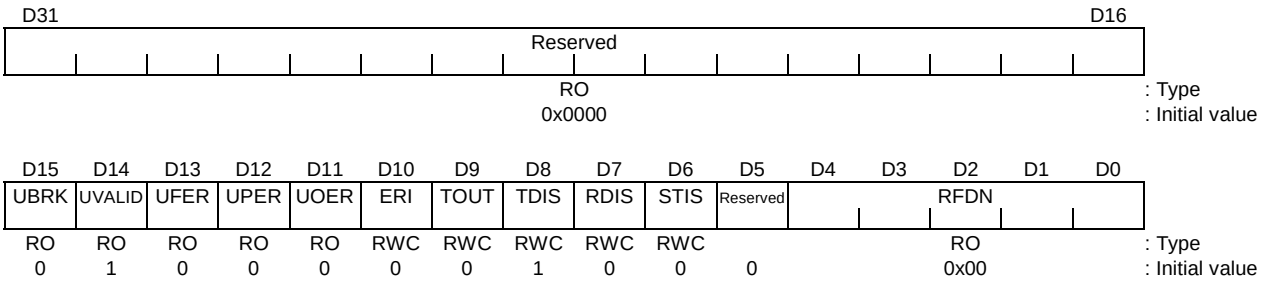
Bits	Mnemonic	Field Name	Description
D31:D14		Reserved	
D13	TIE	Transmit Data Empty Interrupt Enable	Transmit Data Empty Interrupt Enable Transmit Data Empty Interrupt Enable (Default: 0) When there is open space in the Transmit FIFO, this field sets whether to signal an interrupt. 0: Do not signal an interrupt when there is open space in the Transmit FIFO. 1: Signal an interrupt when there is open space in the Transmit FIFO.
D12	RIE	Reception Data Full Interrupt Enable	Reception Data Full Interrupt Enable Receive Data Full Interrupt Enable (Default: 0) This field sets whether to signal interrupts when reception data is full (SIDISRn.RDIS = 1) or a reception time out (SIDISRn.TOUT = 1) occurs. 0: Do not signal interrupts when reception data is full/reception time out occurred. 1: Signal interrupts when reception data is full/reception time out occurred.
D11	SPIE	Reception Error Interrupt Enable	Reception Error Interrupt Enable Receive Data Error Interrupt Enable (Default: 0) This field sets whether to signal interrupts when a reception error (Frame Error, Parity Error, Overrun Error) occurs (SIDISR.ERI = 1). 0: Do not signal reception error interrupts. 1: Signal reception error interrupts.
D10:D9	CTSAC	CTSS Active Condition	CTSS Active Condition CTSS Active Condition (Default: 00) This field specifies status change interrupt request conditions using the CTS Status (CTSS) of the Status Change Interrupt Status Register. 00: Do not detect CTS signal changes. 01: Rising edge of the CTS pin 10: Falling edge of the CTS pin 11: Both edges of the CTS pin
D8:D6		Reserved	

Figure 9.2.36 Interrupt Control Register (1/2)

Bits	Mnemonic	Field Name	Description
D5:D0	STIE	Status Change Interrupt Enable	Status Change Interrupt Enable Status Change Interrupt Enable (Default: 0x00) This field sets the set conditions of the Status Change bit (STIS) of the Interrupt Status Register (SIDISR). The condition is selected depending on which bit of the Status Change Interrupt Status Register (SISCISR) is set. (Multiple selections arepossible.) An SIO interrupt is asserted when STIC is “1”. 000000: Do not detect status changes. 1*****: Set “1” to STIS when the Overrun bit (OERS) is “1”. *1****: Set “1” to STIS when a change occurs in a condition set by the CTSS Active Condition field (CTSAC) in the CTS Status bit (CTSS). **1***: Set “1” to STIS when the Break bit (RBRKD) becomes “1”. ***1**: Set “1” to STIS when the Transmit Data Empty bit (TRDY) becomes “1”. ****1*: Set “1” to STIS when the Transmission Complete bit (TXALS) becomes “1”. *****1: Set “1” to STIS when the Break Detection bit (UBRKD) becomes “1”.

Figure 9.2.36 Interrupt Control Register (2/2)

9.2.4.3 Interrupt Status Register (SIDISR, offset 0x08-0x0B)



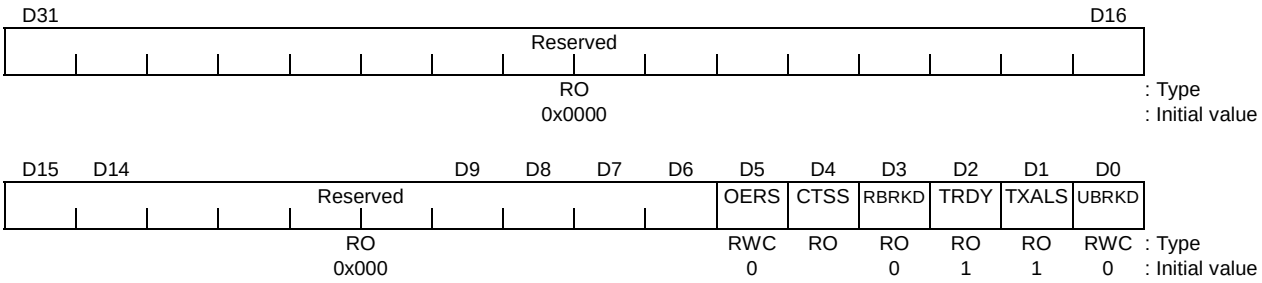
Bits	Mnemonic	Field Name	Description
D31:D16		Reserved	
D15	UBRK	Receive Break	Receive Break UART Break (Default: 0) This field indicates the break reception status of the next data in the Receive FIFO to be read. Reading the Receive FIFO Register (SIRFIFO) updates the status. 0: No breaks 1: Detect breaks
D14	UVALID	Receive FIFO Available Status	Receive FIFO Available Status UART Available Data (Default: 1) This field indicates whether or not data exists in the Receive FIFO (SIRFIFO). 0: Data exists in the Receive FIFO. 1: No data exists in the Receive FIFO.
D13	UFER	Frame Error	Frame Error UART Frame Error (Default: 0) This field indicates the frame error status of the next data in the Receive FIFO to be read. Reading the Receive FIFO Register (SIRFIFO) updates the status. 0: There are no frame errors. 1: There are frame errors.
D12	UPER	Parity Error	Parity Error UART Parity Error (Default: 0) This field indicates the parity error status of the next data in the Receive FIFO to be read. Reading the Receive FIFO Register (SIRFIFO) updates the status. 0: There are no parity errors. 1: There are parity errors.
D11	UOER	Overrun Error	Overrun Error UART Overrun Error (Default: 0) This register indicates the overrun status of the next data in the Receive FIFO to be read. Reading the Receive FIFO Register (SIRFIFO) updates the status. 0: There are no overrun errors. 1: There are overrun errors.

Figure 9.2.37 Interrupt Status Register (1/2)

Bits	Mnemonic	Field Name	Description
D10	ERI	Reception Error Interrupt	Reception Error Interrupt Receive Data Error Interrupt (Default: 0) This bit is immediately set to “1” when a reception error (Frame Error, Parity Error, or Overrun Error) is detected. This bit is cleared when a “0” is written.
D9	TOUT	Reception Time Out	Reception Time Out Time Out (Default: 0) This bit is set to “1” when a reception time out occurs. This bit is cleared when a “0” is written.
D8	TDIS	Transmission Data Empty	Transmission Data Empty Transmit Interrupt Status (Default: 1) This bit is set when available space of the amount set by the Transmit FIFO Request Trigger Level (TDIL) of the FIFO Control Register (SIFCR) exists in the Transmit FIFO. This bit is cleared when a “0” is written.
D7	RDIS	Reception Data Full	Reception Data Full Receive Interrupt Status (Default: 0) This bit is set when valid data of the amount set by the Receive FIFO Request Trigger Level (RDIL) of the FIFO Control register (SIFCR) is stored in the Receive FIFO. This bit is cleared when a “0” is written.
D6	STIS	Status Change	Status Change Status Change Interrupt Status (Default: 0) This bit is set when at least one of the interrupt statuses selected by the Status Change. Interrupt Condition field (STIE) of the Interrupt Control Register (SIDICR) becomes“1”. This bit is cleared when a “0” is written.
D5		Reserved	
D4:D0	RFDN	Reception Data Stage Status	Reception Data Stage Status Receive FIFO Data Number (Default: 00000) This field indicates how many stages of reception data remain in the Receive FIFO (0-16 stages).

Figure 9.2.37 Interrupt Status Register (2/2)

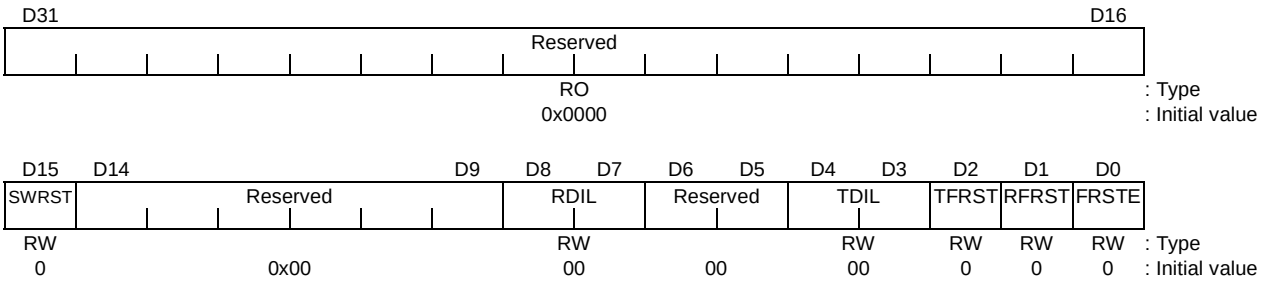
9.2.4.4 Status Change Interrupt Status Register (SISCISR, offset 0x0C-0x0F)



Bits	Mnemonic	Field Name	Description
D31:D6		Reserved	
D5	OERS	Overrun Error	Overrun Error This bit is immediately set to “1” when an overrun error is detected. This bit is cleared when a “0” is written.
D4	CTSS	CTS Status	CTS Status CTS Terminal Status (Default: 1) This field indicates the status of the CTS signal. 1: The CTS signal is High. 0: The CTS signal is Low.
D3	RBRKD	Receiving Break	Receiving Break Receive Break (Default: 0) This bit is set when a break is detected. This bit is automatically cleared when a frame that is not a break is received. 1: Current status is Break. 0: Current status is not Break.
D2	TRDY	Transmission Data Empty	Transmission Data Empty Transmit Ready (Default: 1) This bit is set to “1” if at least one stage in the Transmit FIFO is free.
D1	TXALS	Transmission Complete	Transmission Complete Transmit All Sent (Default: 1) This bit is set to “1” if the Transmit FIFO and all transmission shift registers are empty.
D0	UBRKD	Break Detected	Break Detected UART Break Detect (Default: 0) This bit is set when a break is detected. Once set, this bit remains set until cleared by writing a “0” to it.

Figure 9.2.38 Status Change Interrupt Status Register

9.2.4.5 FIFO Control Register (SIFCR, offset 0x10-0x13)



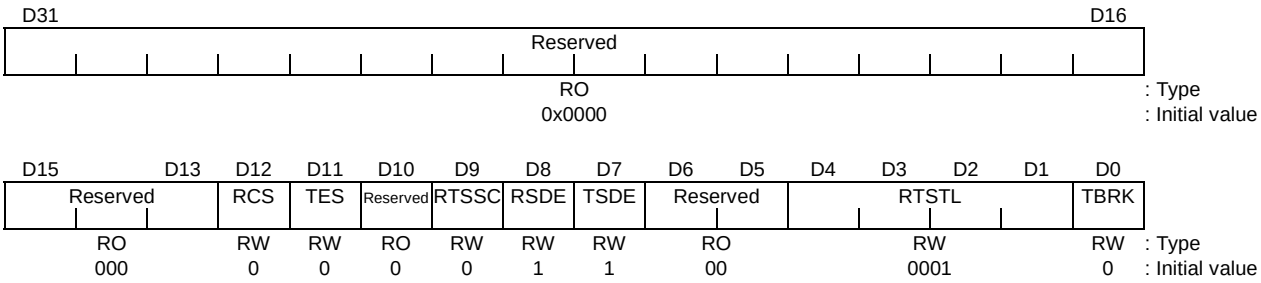
Bits	Mnemonic	Field Name	Description
D31:D16		Reserved	
D15	SWRST	Software Reset	Software Reset Software Reset (Default: 0) This field performs SIO resets except for the FIFOs. Setting this bit to “1” initiates the reset. Set registers are also initialized. This bit returns to “0” when initialization iscomplete. 0: Normal operation 1: SIO software reset
D14:D9		Reserved	
D8:D7	RDIL	Receive FIFO Request Trigger Level	Receive FIFO Request Trigger Level Receive FIFO Interrupt Trigger Level (Default: 00) This register sets the level for reception data transfer from the Receive FIFO. 00: 1 Byte 01: 4 Bytes 10: 8 Bytes 11: 12 Bytes
D6:D5		Reserved	
D4:D3	TDIL	Transmit FIFO Request Trigger Level	Transmit FIFO Request Trigger Level Transmit FIFO Interrupt Trigger Level (Default: 00) This register sets the level for transmission data transfer to the Transmit FIFO. 00: 1 Byte 01: 4 Bytes 10: 8 Bytes 11: Setting disabled
D2	TFRST	Transmit FIFO Reset	Transmit FIFO Reset Transmit FIFO Reset (Default: 0) The Transmit FIFO buffer is reset when this bit is set. This bit is valid when the FIFO Reset Enable bit (FRSTE) is set. Cancel reset by using the software to clear this bit. 0: During operation 1: Reset Transmit FIFO

Figure 9.2.39 FIFO Control Register (1/2)

Bits	Mnemonic	Field Name	Description
D1	RFRST	Receive FIFO Reset	Receive FIFO Reset Receive FIFO Reset (Default: 0) The Receive FIFO buffer is reset when this bit is set. This bit is valid when the FIFO Reset Enable bit (FRSTE) is set. Cancel reset by using the software to clear this bit. 0: During operation 1: Reset Receive FIFO
D0	FRSTE	FIFO Reset Enable	FIFO Reset Enable FIFO Reset Enable (Default: 0) This field is the Reset Enable for the Transmit/Receive FIFO buffer. The FIFO is reset by combining the Transmit FIFO Reset bit (TFRST) and Receive FIFO Reset bit (RRST). 0: During operation 1: Reset Enable

Figure 9.2.39 FIFO Control Register (2/2)

9.2.4.6 Flow Control Register (SIFLCR, offset 0x14-0x17)



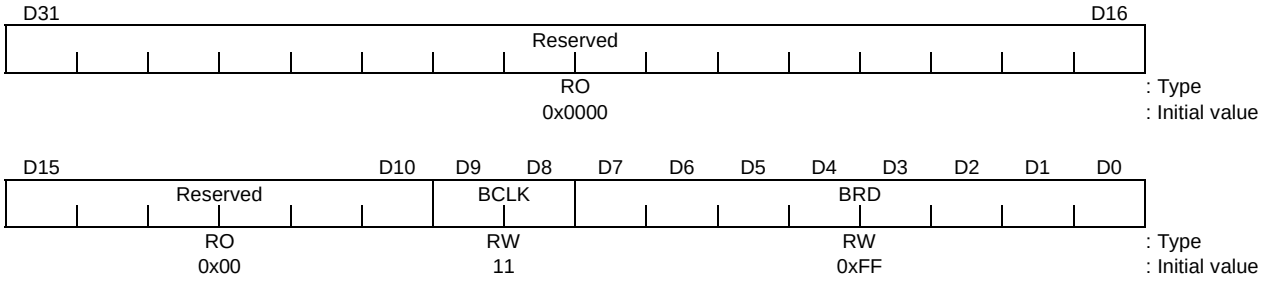
Bits	Mnemonic	Field Name	Description
D31:D13		Reserved	
D12	RCS	RTS Signal Control Select	RTS Signal Control Select RTS Control Select (Default: 0) This field sets the reception flow control using RTS output signals. 0: Disable flow control using RTS signals. 1: Enable flow control using RTS signals.
D11	TES	CTS Signal Control Select	CTS Signal Control Select CTS Control Select (Default: 0) This field sets the transmission flow control using CTS input signals. 0: Disable flow control using CTS signals. 1: Enable flow control using CTS signals.
D10		Reserved	
D9	RTSSC	RTS Software Control	RTS Software Control RTS Software Control (Default: 0) This register is used for software control of RTS output signals. 0: Set the RTS signal to Low (can receive data). 1: Sets the RTS signal to High (transmission pause request)
D8	RSDE	Serial Data Reception Enable	Serial Data Reception Enable Receive Serial Data Enable (Default: 1) This is the Serial Data Enable bit. When this bit is cleared, data reception starts after the start bit is detected. The RTS signal will not become High even if this bit is cleared. 0: Enable (can receive data) 1: Disable (halt reception)
D7	TSDE	Serial Data Transmit Enable	Serial Data Transmit Enable Transmit Serial Data Enable (Default: 1) This is the Serial Data Transmission Enable bit. When this bit is cleared, data transmission starts. When set, transmission stops after completing transmission ofthe current frame. 0: Enable (can transmit data) 1: Disable (halt transmission)
D6:D5		Reserved	

Figure 9.2.40 Flow Control Register (1/2)

Bits	Mnemonic	Field Name	Description
D4:D1	RTSTL	RTS Active Trigger Level	RTS Active Trigger Level RTS Trigger Level (Default: 0001) The RTS hardware control assert level is set by the reception data stage count of the Receive FIFO. 0000: Disable setting 0001: 1 : 1111: 15
D0	TBRK	Break Transmission	Break Transmission Break Transmit (Default: 0) Transmits a break. The TXD signal is Low while TBRK is set to “1”. 0: Disable (clear break) 1: Enable (transmit break)

Figure 9.2.40 Flow Control Register (2/2)

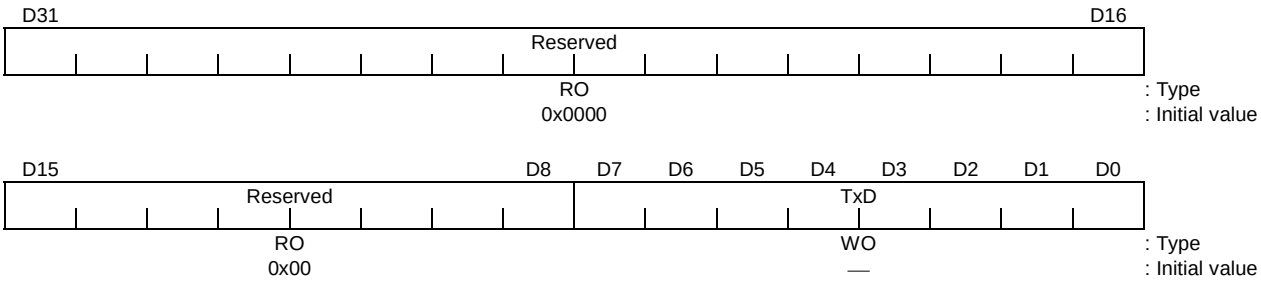
9.2.4.7 Baud Rate Control Register (SIBGR, offset 0x18-0x1B)



Bits	Mnemonic	Field Name	Description
D31:D10		Reserved	
D9:D8	BCLK	Baud Rate Generator Clock	Baud Rate Generator Clock Baud Rate Generator Clock (Default: 11) This field sets the input clock for the baud rate generator. 00: Select prescalar output T0 (fc/2) 01: Select prescalar output T2 (fc/8) 10: Select prescalar output T4 (fc/32) 11: Select prescalar output T6 (fc/128)
D7:D0	BRD	Baud Rate Divide Value	Baud Rate Divide Value Baud Rate Divide Value (Default: 0xFF) This field set divide value BRG of the baud rate generator. This value is expressed as a binary value.

Figure 9.2.41 Baud Rate Control Register

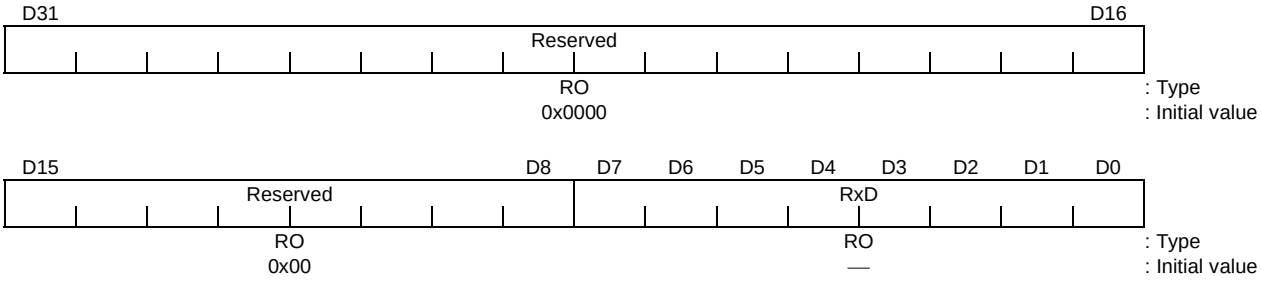
9.2.4.8 Transmit FIFO Register (SITFIFO, offset 0x1C-0x1F)



Bits	Mnemonic	Field Name	Description
D31:D8		Reserved	
D7:D0	TxD	Transmission Data	Transmission Data Transmit Data Data written to this register are written to the Transmit FIFO.

Figure 9.2.42 Transmit FIFO Register

9.2.4.9 Receive FIFO Register (SIRFIFO, offset 0x20-0x23)



Bits	Mnemonic	Field Name	Description
D31:D8		Reserved	
D7:D0	RxD	Reception Data	Reception Data Receive Data This field reads reception data from the Receive FIFO. Reading this register updates the Reception Data Status.

Figure 9.2.43 Receive FIFO Register

9.2.5 GPIO I/O Registers

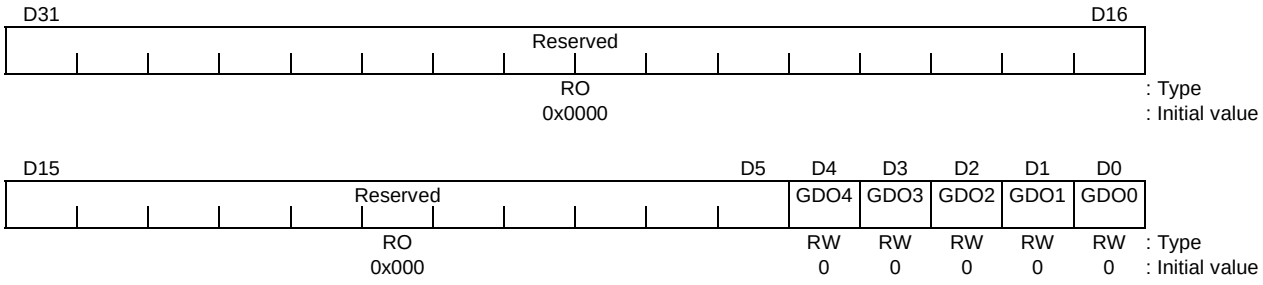
This register is accessed according to the offset from the base address specified by the GPIO I/O Base Address Register of the PCI Configuration Register.

GPIO, SIO and I2C are Multiplexed each pins. Their functions can select by Mode Select Resister in the this module.

Table 9.2.4 GPIO I/O Register List

Offset Address	Size	Register	R/W
0x00	4	GPIO Data Out Register (GPIODO)	RW
0x04	4	GPIO Pin Data Register (GPIODI)	RO
0x08	4	GPIO Direction Control Register (GPIODIR)	RW
0x0C	4	GPIO Open Drain Control Register (GPIOOD)	RW
0x10	4	GPIO Rising Edge Detection Register (GPRER)	RW
0x14	4	GPIO Falling Edge Detection Register (GPFER)	RW
0x18	4	GPIO Interrupt Flag Register (GPINTFR)	RWC
0x1C	4	GPIO Interrupt Enable Register (GPINTER)	RW
0x20	4	Mode Select Register	RW
0x24-0x3F	0x1C	Reserved (fixed to “0”)	RO

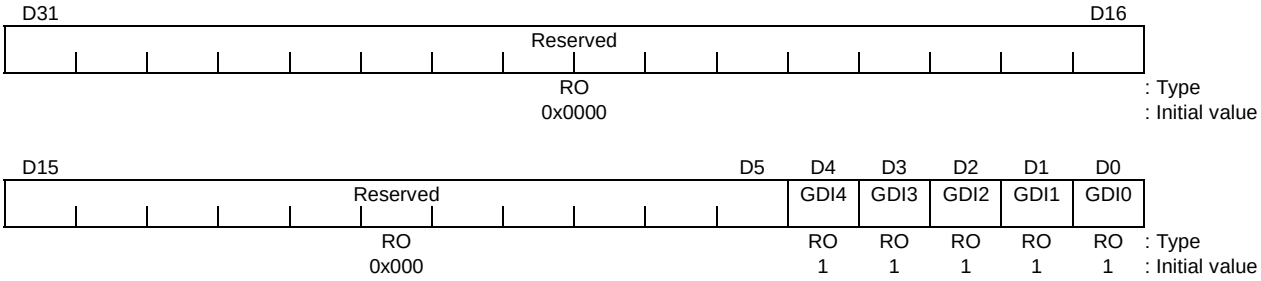
9.2.5.1 GPIO Data Out Register (GPIODO, offset 0x00-0x03)



Bits	Mnemonic	Field Name	Description
D31:D5		Reserved	
D4:D0	GDO	Data Out	Data Out Writes data to GPIO pins. D4 to GPIO4, …, D0 to GPIO0

Figure 9.2.44 GPIO Data Out Register

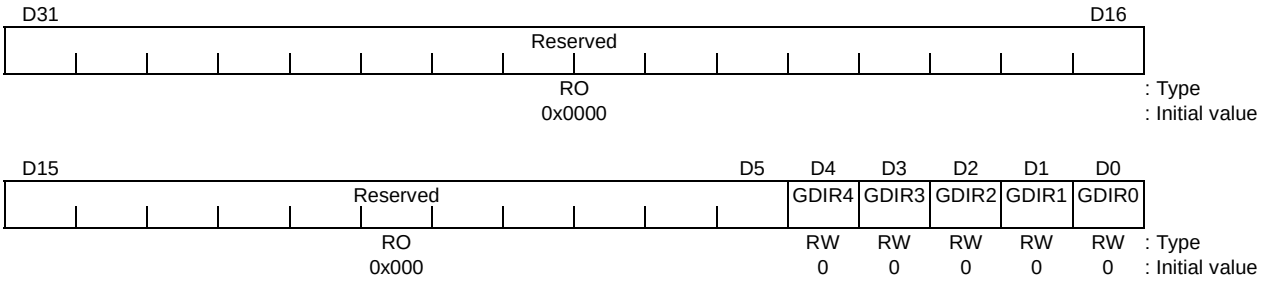
9.2.5.2 GPIO Pin Data Register (GPIODI, offset 0x04-0x07)



Bits	Mnemonic	Field Name	Description
D31:D5		Reserved	
D4:D0	GDI	Pin Data	Pin Data Reads GPIO pins. GPIO4 to D4, … , GPIO0 to D0

Figure 9.2.45 GPIO Pin Data Register

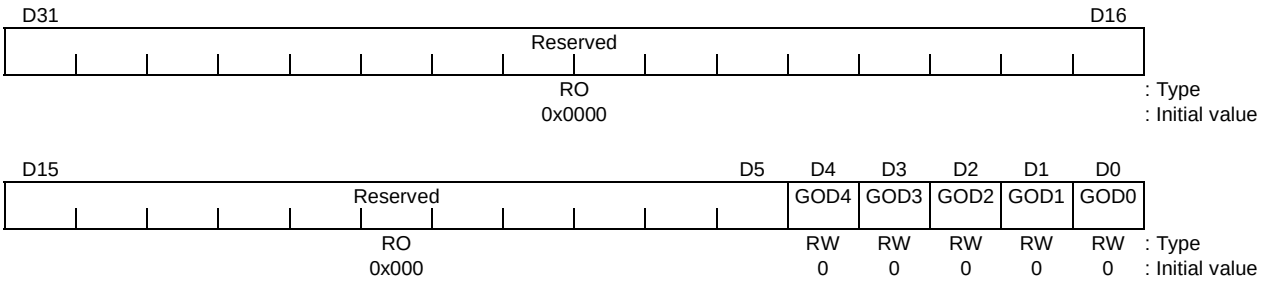
9.2.5.3 GPIO Direction Control Register (GPIODIR, offset 0x08-0x0B)



Bits	Mnemonic	Field Name	Description
D31:D5		Reserved	
D4:D0	GDIR	Direction Control	Direction Control 0: Input (Reset) 1: Output

Figure 9.2.46 GPIO Direction Control Register

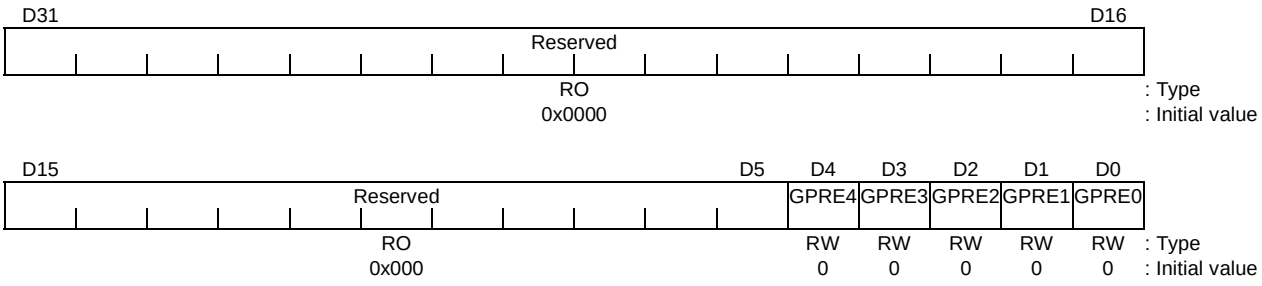
9.2.5.4 GPIO Open Drain Control Register (GPIOOD, offset 0x0C-0x0F)



Bits	Mnemonic	Field Name	Description
D31:D5		Reserved	
D4:D0	GOD	Open Drain Control	Open Drain Control 0: Open-drain (Reset) 1: Totem-pole

Figure 9.2.47 GPIO Open Drain Control Register

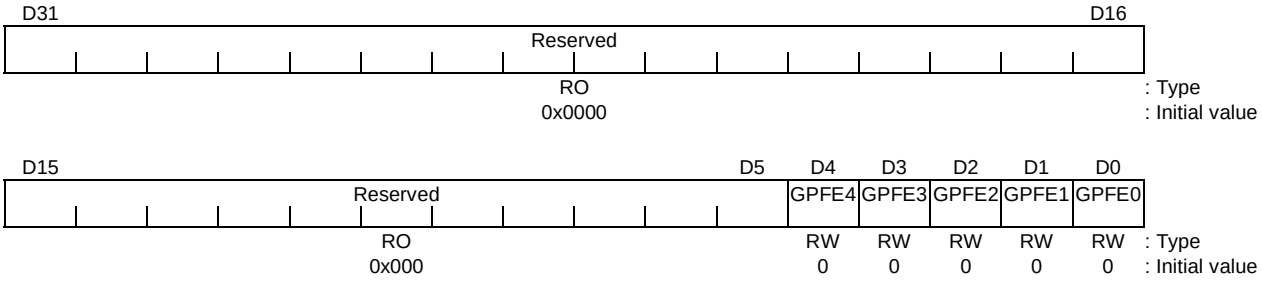
9.2.5.5 GPIO Rising Edge Detection Register (GPRER, offset 0x10-0x13)



Bits	Mnemonic	Field Name	Description
D31:D5		Reserved	
D4:D0	GPRE	Rising Edge Detection	Rising Edge Detection Detects rising edges. 0: Does not detect rising edges. 1: Sets a “1” in the Interrupt Flag Register when detecting a rising edge.

Figure 9.2.48 GPIO Rising Edge Detection Register

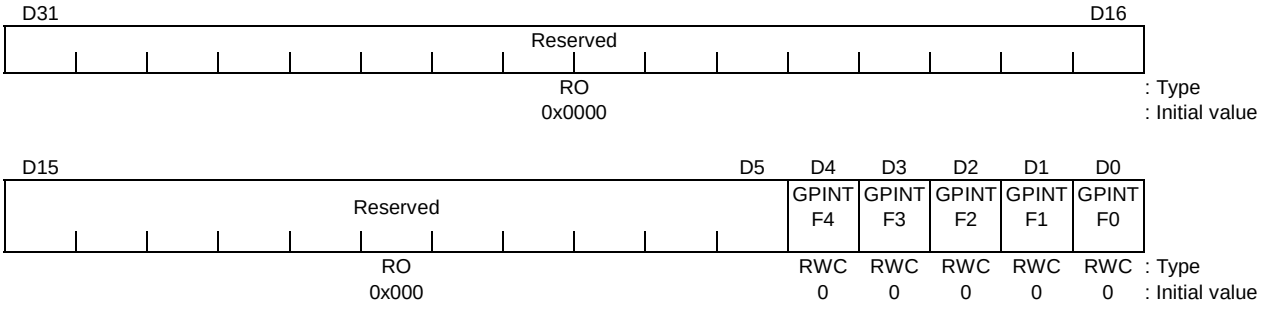
9.2.5.6 GPIO Falling Edge Detection Register (GPFER, offset 0x14-0x17)



Bits	Mnemonic	Field Name	Description
D31:D5		Reserved	
D4:D0	GPFE	Falling Edge Detection	Falling Edge Detection Detects falling edges. 0: Does not detect falling edges. 1: Sets a “1” in the Interrupt Flag Register when detecting a falling edge.

Figure 9.2.49 GPIO Falling Edge Detection Register

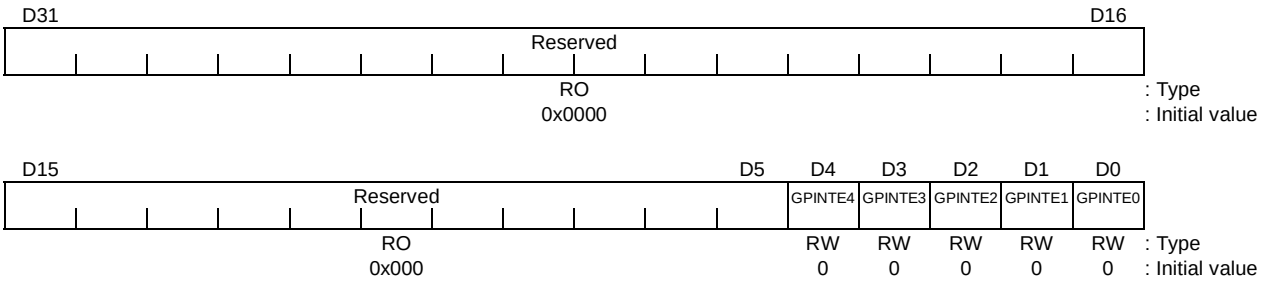
9.2.5.7 GPIO Interrupt Flag Register (GPINTFR, offset 0x18-0x1B)



Bits	Mnemonic	Field Name	Description
D31:D5		Reserved	
D4:D0	GPINTF	Interrupt Flag	Interrupt Flag Interrupt Request Flag Read 0 Interrupt request cancellation state 1 Interrupt state generation state Write 0 No change 1 Interrupt request canceled

Figure 9.2.50 GPIO Interrupt Flag Register

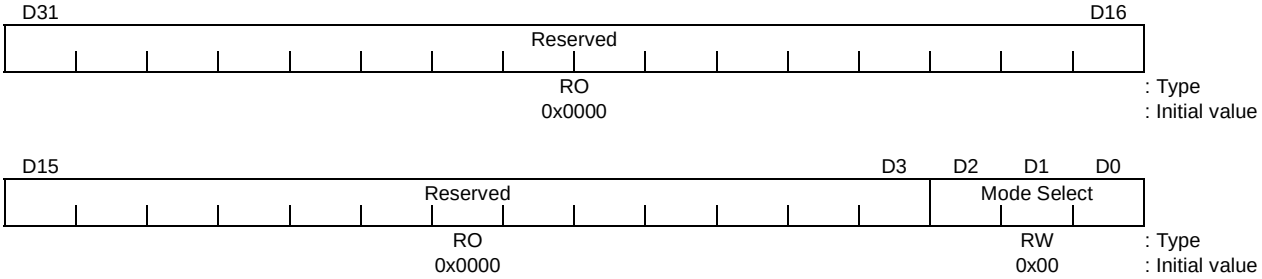
9.2.5.8 GPIO Interrupt Enable Register (GPINTER, offset 0x1C-0x1F)



Bits	Mnemonic	Field Name	Description
D31:D5		Reserved	
D4:D0	GPINTE	Interrupt Enable	Interrupt Enable 0: Interrupt mask state 1: Interrupt Flag Register outputs interrupt signals

Figure 9.2.51 GPIO Interrupt Enable Register

9.2.5.9 Mode Select Register (MODESELR, offset 0x20-0x23)



Bits	Mnemonic	Field Name	Description
D31:D3		Reserved	
D2:D0	Mode Select	Mode Select	Mode Select 000: Mode0: GPIO 5pin (0, 1, 2, 3, 4) 001: Mode1: I²C 1ch, GPIO 3pin (0, 1, 2) 010: Mode2: SIO 1ch (Internal clock),GPIO 3pin (0, 3, 4) 011: Mode3: I²C 1ch, SIO 1ch (Internal clock),GPIO 1pin (0) 100: Mode4: I²C 1ch, SIO 1ch (External clock) 101: Mode5: SIO 1ch (Internal clock), GPIO 1pin (0) 110: Mode6: SIO 1ch (External clock) 111: Reserved Mode0: GPIO4 GPIO3 GPIO2 GPIO1 GPIO0 Mode1: SDA SCL GPIO2 GPIO1 GPIO0 Mode2: GPIO4 GPIO3 TXD RXD GPIO0 Mode3: SDA SCL TXD RXD GPIO0 Mode4: SDA SCL TXD RXD SCLK Mode5: RTS* CTS* TXD RXD GPIO0 Mode6: RTS* CTS* TXD RXD SCLK

Figure 9.2.52 Mode Select Register

9.2.6 Internal G-bus Arbiter I/O Register Map

These registers are accessed by referring to the Internal G-bus Arbiter I/O register base address that is set by a configuration register as an offset to the accessed registers. The I/O register base address must first be set before accessing these registers.

9.2.6.1 G-bus Arbiter Setting Register (0x00-0x01)

D15	...	D7	D6	D5	D4	D3	D2	D1	D0	
Reserved			IBRE	PRI4	PRI3	PRI2	PRI1	PRI0	FRS	
RO			RW	RW	RW	RW	RW	RW	RW	: Type
0x000			1	0	0	0	0	0	0	: Initial value

Bits	Mnemonic	Field Name	Description
D15:D7		Reserved	
D6	IBRE	IO BUS Request Enable	Arbitration for Target DMA Read IO cycle and IO Write Pending cycles to the ATA and I²C/SIO/GPIO(PBUS). 0:Disable 1:Enable
D5:D1	PRI	Fixed priority Setting	Fixed priority Setting

D5 D4 D3 D2 D1	Priority High ← → Priority Low			
0 0 0 0 0	ATA	USB Host	USB Device	PCI Target
0 0 0 0 1	ATA	USB Host	PCI Target	USB Device
0 0 0 1 0	ATA	USB Device	USB Host	PCI Target
0 0 0 1 1	ATA	USB Device	PCI Target	USB Host
0 0 1 0 0	ATA	PCI Target	USB Host	USB Device
0 0 1 0 1	ATA	PCI Target	USB Device	USB Host
0 0 1 1 0	USB Host	ATA	USB Device	PCI Target
0 0 1 1 1	USB Host	ATA	PCI Target	USB Devicet
0 1 0 0 0	USB Host	USB Device	ATA	PCI Target
0 1 0 0 1	USB Host	USB Device	PCI Target	ATA
0 1 0 1 0	USB Host	PCI Target	ATA	USB Device
0 1 0 1 1	USB Host	PCI Target	USB Device	ATA
0 1 1 0 0	USB Device	ATA	USB Host	PCI Target
0 1 1 0 1	USB Device	ATA	PCI Target	USB Host
0 1 1 1 0	USB Device	USB Host	ATA	PCI Target
0 1 1 1 1	USB Device	USB Host	PCI Target	ATA
1 0 0 0 0	USB Device	PCI Target	ATA	USB Host
1 0 0 0 1	USB Device	PCI Target	USB Host	ATA
1 0 0 1 0	PCI Target	ATA	USB Host	USB Device
1 0 0 1 1	PCI Target	ATA	USB Device	USB Host
1 0 1 0 0	PCI Target	USB Host	ATA	USB Device
1 0 1 0 1	PCI Target	USB Host	USB Device	ATA
1 0 1 1 0	PCI Target	USB Device	ATA	USB Host
1 0 1 1 1	PCI Target	USB Device	USB Host	ATA
1 1 0 0 0		Reserved		
1 1 0 0 1		Reserved		
1 1 0 1 0		Reserved		
1 1 0 1 1		Reserved		
1 1 1 0 0		Reserved		
1 1 1 0 1		Reserved		
1 1 1 1 0		Reserved		
1 1 1 1 1		Reserved		

D0	FRS	FRS	Arbiter mode select 0: Round Robin 1: Fixed

Figure 9.2.53 G-bus Arbiter Setting Register

10. Loadable PCI Configuration Space

GOKU-S PCI core design includes a Loadable Configuration Space.

This feature enables core on power-up to load the read only configuration registers with its PCI Configuration Space from a standard 93C57 (2Kbit) Serial EEPROM. Information such as Vender ID, which is typically hard-coded into the chip containing the core, can instead be loaded at system start up, allowing the same chip to be used in more than one system. GOKU-S only supports x16 memory organization.

Immediately after hardware power-on reset, the core reads the EEPROM and loads the read-only configuration registers sequentially from the first 64 bytes in the EEPROM.

10.1 EEPROM interface Signals

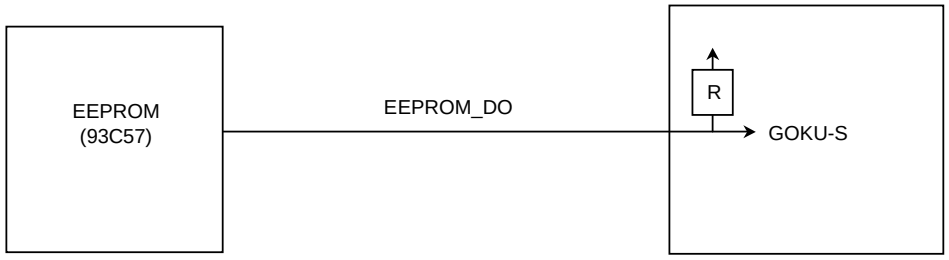
The PCI core's application interface contains the following signals to support this capability.

Table 10.1.1 EEPROM Interface Signals for 93C57

Signal	Dir	Pin No.	Function
EEPROM_DI	OUT	92	Data to the EEPROM from the core
EEPROM_DO	IN	87	Data from the EEPROM to the core
EEPROM_CS	OUT	86	Chip select
EEPROM_SK	OUT	95	Serial clock

10.2 How to Disable of Serial EEPROM

If Loadable PCI Configuration Space function is disable, EEPROM_DO signal would always be connected to VDD due to internal pull-up resister.



10.3 Image of PCI Configuration Space Mapping

The EEPROM must be loaded as shown in Table 10.3.1 with an Image of the PCI Configuration Space data that will be read by the BIOS software. Only the fields defined as " read-only" in the PCI 2.1 Specification should contain valid data. The read-only fields are indicated by name in the table, and should contain valid data.

Table 10.3.1 PCI Configuration Space Mapping Image

23	15	07	00	
Device ID		Vendor ID		0x00
	Cfg04_bit_4_5			0x04
Class Code (High)		Class Code (Low)	Revision ID	0x08
	Header			0x0C
				0x10
				0x14
				0x18
				0x1C
				0x20
				0x24
				0x28
Sub System ID		Sub Vendor ID		0x2C
				0x30
			Capability Pointer	0x34
				0x38
Maximum Latency	Minimum GNT	Interrupt Pin		0x3C
				0x40-0xD8
PME support, D1D2 support, Aux Current, PME clock		Next Item Pointer		0xDC
Data	B2 / B3 support for D3 hot	Data Scale, Data Select		0xE0
				0xE4-0xF8
				0xFC

EEPROM Image

ROM Addresss	FIELD NAME	DEFAULT (2 byte width)	Valid bits	EEPROM Loadable	EEPROM VALUE USED	EEPROM VALUE USED			
						ATA	USB Host	USB Device	I2C/SIO/GPIO
0x00	Number of Function	0x0004		YES	YES	EEPROM value	EEPROM value	EEPROM value	EEPROM value
0x01	Vender_ID	0x102F	[15:0]	YES	YES	EEPROM value	EEPROM value	EEPROM value	EEPROM value
0x02	Device_ID	FIXED	[15:0]	YES	NO	0x0105	0x0106	0x0107	0x0108
0x03	Cfg04_bit_4_5	0x0001	[1:0]	YES	YES	EEPROM value	EEPROM value	EEPROM value	EEPROM value
0x04	Revision_ID	FIXED	[7:0]	YES	NO	00	00	00	00
0x05	Subvender_ID	0x0000	[15:0]	YES	YES	EEPROM value	EEPROM value	EEPROM value	EEPROM value
0x06	Subsystem_ID	0x0000	[15:0]	YES	YES	EEPROM value	EEPROM value	EEPROM value	EEPROM value
Function 0 (ATA)									
0x07	Function _ID_0	0x0000	[7:0]	YES	YES	Must set to 0x0000			
0x08	Class_code_l_0	0x0085	[7:0]	YES	YES				
0x09	Class_code_h_0	0x0101	[15:0]	YES	YES				
0x0A	Header_0	0x0080	[7:0]	YES	YES				
0x0B	Int_pin_0	0x0001	[7:0]	YES	YES				
0x0C	Reg3c_0 (max_lat/min_gnt)	0x0804	[15:0]	YES	YES				
0x0D	Cap_pointer_0	0x00DC	[7:0]	YES	YES				
0x0E	Next_item_ptr_0	0x0000	[7:0]	YES	YES				
0x0F	Pme_support_0	0x0000	[4:0]	YES	YES				
0x10	D1d2_support_0	0x0000	[1:0]	YES	YES				
0x11	Aux_current_0	0x0000	[2:0]	YES	YES				
0x12	Pmeclk_0	0x0000	[1]	YES	YES				
0x13	Data_scale_0	0x0000	[1:0]	YES	YES				
0x14	Data_select_0	0x0000	[3:0]	YES	YES				
0x15	Bpcc_b2b3_0	0x0000	[1:0]	YES	YES				
0x16	Data_reg_0	0x0000	[7:0]	YES	YES				
Function 1 (USB Host)									
0x17	Function _ID_1	0x0001	[7:0]	YES	YES	Must set to 0x0001			
0x18	Class_code_l_1	0x0010	[7:0]	YES	YES				
0x19	Class_code_h_1	0x0C03	[15:0]	YES	YES				
0x1A	Header_1	0x0080	[7:0]	YES	YES				
0x1B	Int_pin_1	0x0001	[7:0]	YES	YES				
0x1C	Reg3c_1 (max_lat/min_gnt)	0x2B01	[15:0]	YES	YES				
0x1D	Cap_pointer_1	0x00DC	[7:0]	YES	YES				

ROM Address	FIELD NAME	DEFAULT (2 byte width)	Valid bits	EEPROM Loadable	EEPROM VALUE USED	EEPROM VALUE USED			
						ATA	USB Host	USB Device	I2C/SIO/GPIO
0x1E	Next_item_ptr_1	0x0000	[7:0]	YES	YES				
0x1F	Pme_support_1	0x0000	[4:0]	YES	YES				
0x20	D1d2_support_1	0x0000	[1:0]	YES	YES				
0x21	Aux_current_1	0x0000	[2:0]	YES	YES				
0x22	Pmeclk_1	0x0000	[1]	YES	YES				
0x23	Data_scale_1	0x0000	[1:0]	YES	YES				
0x24	Data_select_1	0x0000	[3:0]	YES	YES				
0x25	Bpcc_b2b3_1	0x0000	[1:0]	YES	YES				
0x26	Data_reg_1	0x0000	[7:0]	YES	YES				
Function 2 (USB Device)									
0x27	Function _ID_2	0x0002	[7:0]	YES	YES	Must set to 0x0002			
0x28	Class_code_l_2	0x00FE	[7:0]	YES	YES				
0x29	Class_code_h_2	0x0C03	[15:0]	YES	YES				
0x2A	Header_2	0x0080	[7:0]	YES	YES				
0x2B	Int_pin_2	0x0001	[7:0]	YES	YES				
0x2C	Reg3c_2 (max_lat/min_gnt)	0x2B01	[15:0]	YES	YES				
0x2D	Cap_pointer_2	0x00DC	[7:0]	YES	YES				
0x2E	Next_item_ptr_2	0x0000	[7:0]	YES	YES				
0x2F	Pme_support_2	0x0000	[4:0]	YES	YES				
0x30	D1d2_support_2	0x0000	[1:0]	YES	YES				
0x31	Aux_current_2	0x0000	[2:0]	YES	YES				
0x32	Pmeclk_2	0x0000	[1]	YES	YES				
0x33	Data_scale_2	0x0000	[1:0]	YES	YES				
0x34	Data_select_2	0x0000	[3:0]	YES	YES				
0x35	Bpcc_b2b3_2	0x0000	[1:0]	YES	YES				
0x36	Data_reg_2	0x0000	[7:0]	YES	YES				
Function 3 (I2C/SIO/GPIO)									
0x37	Function _ID_3	0x0003	[7:0]	YES	YES	Must set to 0x0003			
0x38	Class_code_l_3	0x0000	[7:0]	YES	YES				
0x39	Class_code_h_3	0x0780	[15:0]	YES	YES				
0x3A	Header_3	0x0080	[7:0]	YES	YES				
0x3B	Int_pin_3	0x0001	[7:0]	YES	YES				
0x3C	Reg3c_3 (max_lat/min_gnt)	0x0000	[15:0]	YES	YES				
0x3D	Cap_pointer_3	0x00DC	[7:0]	YES	YES				
0x3E	Next_item_ptr_3	0x0000	[7:0]	YES	YES				

ROM Address	FIELD NAME	DEFAULT (2 byte width)		EEPROM Loadable	EEPROM VALUE USED	EEPROM VALUE USED			
						ATA	USB Host	USB Device	I2C/SIO/GPIO
0x3F	Pme_support_3	0x0000	[4:0]	YES	YES				
0x40	D1d2_support_3	0x0000	[1:0]	YES	YES				
0x41	Aux_current_3	0x0000	[2:0]	YES	YES				
0x42	Pmeclk_3	0x0000	[1]	YES	YES				
0x43	Data_scale_3	0x0000	[1:0]	YES	YES				
0x44	Data_select_3	0x0000	[3:0]	YES	YES				
0x45	Bpcc_b2b3_3	0x0000	[1:0]	YES	YES				
0x46	Data_reg_	0x0000	[7:0]	YES	YES				

Note: Number of Function (ROM Address =0x00) must be set to 0x0004.

11. JTAG Interface

11.1 JTAG Interface

The GOKU-S JTAG (Joint Test Action Group) Interface provides the IEEE1149.1 standard compliant JTAG Boundary Scan Test.

JTAG Boundary Scan Test

- IEEE1149.1 compatible TAP Controller
- Supports the following six instructions: EXTEST, SAMPLE/PRELOAD, IDCODE, CLAMP, BYPASS, HIGHZ

Pins

JTAG Interface pins are multiplexed with other function signals as follows, and these JTAG Interface pins are enabled when MODE0* and MODE1* and EEPROM_DO are low.

Pin No.	JTAG Interface	Multiplexed signal
75	TRST*	-
86	TDO	EEPROM_CS
58	TDI	USBHOC1*
56	TMS	USBHOC0*
60	TCK	C48MIN

11.2 JTAG Boundary Scan Test

11.2.1 JTAG Controller and Register

The JTAG Interface contains a JTAG Controller (TAP Controller) and a Control Register. This section explains only those portions that are unique to the TC86C001FG (GOKU-S). Please contact your local Toshiba Sales representative for more information regarding the required BSDL files when performing the JTAG Boundary Scan Test.

- Instruction Register (Refer to 11.2.2)
- Data Register
  - Boundary Scan Register (Refer to 11.2.3)
  - Bypass Register
  - Device ID Register (Refer to 11.2.4)
- Test Access Port Controller (TAP Controller) (Refer to 11.3)

11.2.2 Instruction Register

The JTAG Instruction Register consists of a 3-bit shift register. This register is used for selecting either one or both of the test to be performed and the Test Data Register to be accessed. The Data Register is selected according to the instruction code in Table 11.2.1.

Table 11.2.1 Bit Configuration of JTAG Instruction Register

Instruction Code MSB → LSB	Instruction	Selected Data Register
000 (0x00)	EXTEST	Boundary Scan Register
001 (0x01)	IDCODE	Device ID Register
010 (0x02)	SAMPLE/PRELOAD	Boundary Scan Register
011 (0x03)	CLAMP	Bypass Register
100 (0x04)	HIGHZ	Bypass Register
101- 110	Reserved	Reserved
111 (0x07)	BYPASS	Bypass Register

Figure 11.2.1 shows the format of the Instruction Register.

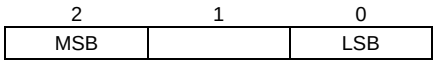


Figure 11.2.1 Instruction Register

The instruction code is shifted to the Instruction Register starting from the Least Significant Bit.

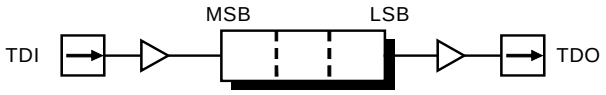


Figure 11.2.2 Shift Direction of the Instruction Register

11.2.3 Boundary Scan Register

The Boundary Scan Register contains a single 179-bit shift register to which all GOKU-S I/O signals except for power supply, TDI, TCK, TDO, TMS, TRST*, MODE[1:0]*, XIN, XOUT, and EEPROM_DO are connected. Figure 11.2.3 shows the bits of the Boundary Scan Register.

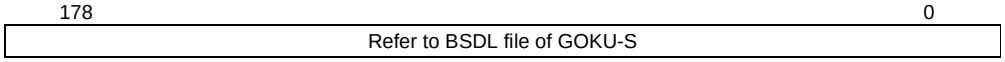


Figure 11.2.3 Boundary Scan Register

TDI input is fetched to the Most Significant Bit (MSB) of the Boundary Scan Register and the Least Significant Bit (LSB) of the Boundary Scan Register is sent from the TDO output.

Table 11.2.2 shows the boundary scan sequence relative to the GOKU-S signals.

Table 11.2.2 TC86C001FG (GOKU-S) JTAG Scan Sequence

JTAG Scan Sequence	Signal Name	JTAG Scan Sequence	Signal Name	JTAG Scan Sequence	Signal Name
		33	AD[5]	66	DD[14]
1	AD[24]	34	AD[4]	67	DD[1]
2	C/BE[3]	35	AD[3]	68	DD[13]
3	IDSEL	36	AD[2]	69	DD[2]
4	AD[23]	37	AD[1]	70	DD[12]
5	AD[22]	38	AD[0]	71	DD[3]
6	AD[21]	39	USBHDP0	72	DD[11]
7	AD[20]	40	USBHDN0	73	DD[4]
8	AD[19]	41	USBHPON0	74	DD[10]
9	AD[18]	42	USBHPON1	75	DD[5]
10	AD[17]	43	GPIO4/SDA/RTS*	76	DD[9]
11	AD[16]	44	GPIO3/SCL/CTS*	77	DD[6]
12	C/BE[2]	45	GPIO2/TXD	78	DD[8]
13	FRAME*	46	GPIO1/RXD	79	DD[7]
14	IRDY*	47	GPIO0/SCLK	80	RESET*
15	TRDY*	48	FULLRST*	81	USBDDP
16	DEVSEL*	49	CGRESET*	82	USBDDN
17	STOP*	50	USBHDP1	83	USBDPULLUP
18	PERR*	51	USBHDN1	84	USBDPDECT
19	SERR*	52	CS1*	85	INTA*
20	PAR	53	CS0*	86	INTB*
21	C/BE[1]	54	DA[2]	87	PCICLK
22	AD[15]	55	DA[0]	88	GNT*
23	AD[14]	56	PDIAG*	89	REQ*
24	AD[13]	57	DA[1]	90	AD[31]
25	AD[12]	58	INTRQ	91	AD[30]
26	AD[11]	59	DMACK*	92	AD[29]
27	AD[10]	60	IORDY	93	AD[28]
28	AD[9]	61	DIOR*	94	AD[27]
29	AD[8]	62	DIOW*	95	AD[26]
30	C_BE[0]	63	DMARQ	96	AD[25]
31	AD[7]	64	DD[15]		
32	AD[6]	65	DD[0]		

11.2.4 Device ID Register

The Device ID Register is a 32-bit shift register. This register is used for reading the ID code that expresses the IC manufacturer, part number, and version from the IC and sending it to a serial device. The following figure shows the configuration of the Device ID Register.

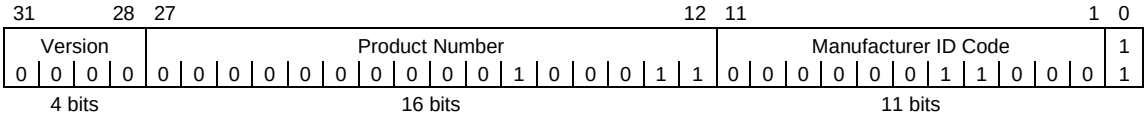


Figure 11.2.4 Device ID Register

The device ID code for the GOKU-S is 0x00023031. However, the four top bits of the Version field may be changed. The device ID code is shifted out from the Least Significant Bit.

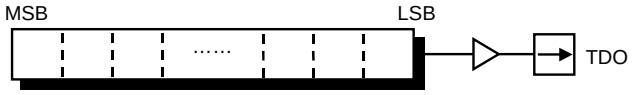


Figure 11.2.5 Shift Direction of the Device ID Register

11.3 Initializing the JTAG Interface

The JTAG Interface is not reset by asserting the FULLRST* signal. Operation of the GOKU-S is not guaranteed if the JTAG Interface is not reset. This interface is initialized by either of the following sequence.

- Set MODE0* and Mode1* and EEPROM_DO to low.
- Assert the TRST* signal.

Externally fix the TRST* signal to GND when not using TRST* for JTAG SCAN.

12. Electrical Characteristics

12.1 Absolute Maximum Rating (1)

Table 12.1.1 Absolute Maximum Rating

Parameter	Symbol	Rating	Unit
Supply Voltage (for I/O)	$V_{CCIOMax}$	−0.3 to 3.9	V
Supply Voltage (for Internal)	$V_{CCINTMax}$	−0.3 to 3.0	V
Input Voltage			
ATA Interface Input Signals	V_{IN1}	−0.3 to 5.8	V
XIN	V_{IN2}	−0.3 to V_{CCInt} + 0.3	V
Other Input Signals (2)	V_{IN3}	−0.3 to V_{CCIO} + 0.3	V
Storage Temperature	T_{STG}	−40 to +125	°C
Power	P_D	0.4	W

(1) If LSI is used above the maximum ratings, permanent destruction of LSI can result. In addition, it is desirable to use LSI for normal operation under the recommended condition. If these conditions are exceeded, reliability of LSI may be adversely affected.

(2) The maximum rated $V_{CCIOMax}$ voltage must not be exceeded even at V_{CCIO} + 0.3 volts.

12.2 Recommended Operating Conditions (3)

Table 12.2.1 Recommended Operating Conditions

Parameter		Symbol	Condition	Min	Max	Unit
Supply Voltage	I/O	V_{CCIO}		3.1	3.5	V
	Internal	V_{CCInt}		1.4	1.6	V
Operating Temperature		Ta		0	85	°C

(3) Functional operation should be restricted to the recommended operating conditions. Those are the limits under which proper device operation is guaranteed. Therefore, the end product must be designed within the recommended voltage and temperature ranges indicated.

12.3 DC Characteristics

12.3.1 DC Characteristics Except for PCI Interface, DP/DN of USB Host/Device, ATA

(Ta = 0 ~ 85 °C, V_{CCIO} = 3.3 V ± 0.2 V, V_{CCInt} = 1.5 V ± 0.1 V, V_{SS} = 0 V)

Parameter	Symbol	Condition	Min	Max	Unit
Low-level Input Voltage	VIL1 VIL1_XIN	(1) XIN	−0.3 −0.3	0.8 0.4	V V
High-level Input Voltage	VIH1 VIH1_XIN	(1) XIN	2.0 1.2	VccIO+0.3 1.6	V V
Low-level Output Current	IOL1	(2) VOL = 0.4 V	4	—	mA
High-level Output Current	IOH1	(2) VOH = 2.4 V	—	−4	mA
Low-level Input Leakage Current	IIL1 IIL2	(3) VIN =VSS (4) VIN =VSS	−10 −200	10 −10	μA μA
High-level Input Leakage Current	IIH1	(5) VIN = VCCIO	−10	10	μA
High -z Output Leakage Current	IOZ	(6)	−10	10	μA
Operating Current (for Internal)	ICCInt			100	mA
Operating Current (for I/O)	ICCIO			76	mA

(1) All input and input-mode bidirectional pins except for PCI interface signals, ATA signals, USB Host/Device signals, XIN.

(2) USBHPON0, USBHPON1, RTS/SDA/GPIO4, CTS/SCL/GPIO3, TXD/GPIO2, RXD/GPIO1, SCLK/GPIO0, EEPROM_CS/TDO, USBPULLUP

(3) USBPON0, USBHPON1, EEPROM_CS/TDO, USBPULLUP

(4) USBHOC0*/TMS, USBHOC1*/TDI, C48MIN/TCK, RTS/SDA/GPIO4, CTS/SCL/GPIO3, TXD/GPIO2, RXD/GPIO1, SCLK/GPIO0, MODE1*, MODE0*, FULLRST*, CGRESET*, EEPROM_DO, TRST*

(5) (3) (4) signals

(6) USBHPON0, USBHPON1

12.3.2 DC Characteristics for PCI Interface

(Ta = 0 ~ 85 °C, V_{CCIO} = 3.3 V ± 0.2 V, V_{CCInt} = 1.5 V ± 0.1 V, V_{SS} = 0 V)

Parameter	Symbol	Conditions	Min	Max	Unit
Low-level Input Voltage	VILPCI	(1)	−0.5	0.9	V
High-level Input Voltage	VIHPCI	(1)	1.8	VccIO+0.3	V
High-level Output Voltage	VOHPCI	(2) IOUT = −500uA	VddIO × 0.9	—	V
Low-level Output Voltage	VOLPCI	(2) IOUT = 1500uA	—	VddIO × 0.1	V
Input Leakage Current	IIHPCI	0 < VIN < VddIO	−10	10	μA
	IILPCI		−10	10	μA

(1) IDSEL, C/BE[3:0], DEVSEL*, FRAME*, GNT*, IRDY*, PAR, AD[31:0], PERR*, REQ*, SERR*, STOP*, TRDY*, INTA*, INTB*

(2) All PCI interface signals except for IDSEL, PCICLK, GNT*

12.3.3 DC Characteristics for ATA Interface

(Ta = 0 ~ 85 °C, V_{CCIO} = 3.3 V ± 0.2 V, V_{CCInt} = 1.5 V ± 0.1 V, V_{SS} = 0 V)

Parameter	Symbol	Conditions	Min	Max	Unit
Low-level Input Voltage	VILATA		0	0.8	V
High-level Input Voltage	VIHATA		2.0	5.5	V
Driver Sink Current	IOL		4		mA
Driver Source Current	IOH		400		μA
Low-level Input Leakage Current	IIL	VIN = VSS	−10	10	μA
High-level Input Leakage Current	IIH	VIN = VCCIO	−10	10	μA
High-level Output Voltage	VOH	(1)	2.4		V
Low-level Output Voltage	VOL	(1)		0.5	V

(1) All ATA Interface signals except for PDIAG*, INTRQ*, IORDY.

12.3.4 DC Characteristics for USB Host/Device

(Ta = 0 ~ 85 °C, V_{CCIO} = 3.3 V ± 0.2 V, V_{CCInt} = 1.5 V ± 0.1 V, V_{SS} = 0 V)

Parameter	Symbol	Conditions	Min	Max	Unit
Input Levels:					
Differential Input Sensitivity	VDI	|(D+) − (D-)|;	0.2		V
Differential Common Mode Range	VCM		0.8	2.5	V
Output Levels:					
Low	VOL	RL of 1.425 kΩ to 3.6 V	0.0	0.3	V
High (Driven)	VOH	RL of 14.25 kΩ to GND	2.8	3.6	V
Output Signal Crossover Voltage	VCRS		1.3	2.0	V

12.4 Crystal Oscillator Characteristics

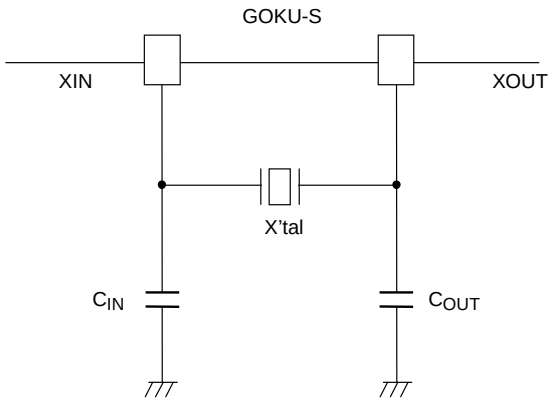


Figure 12.4.1 Electrical Specifications for 8.0 MHz Crystal

Please note that there are some consideration on the location of the external crystal as follows.

(1) Please place the crystal as close to the GOKU-S as possible.

(2) Please place the crystal as far from data bus lines as possible.

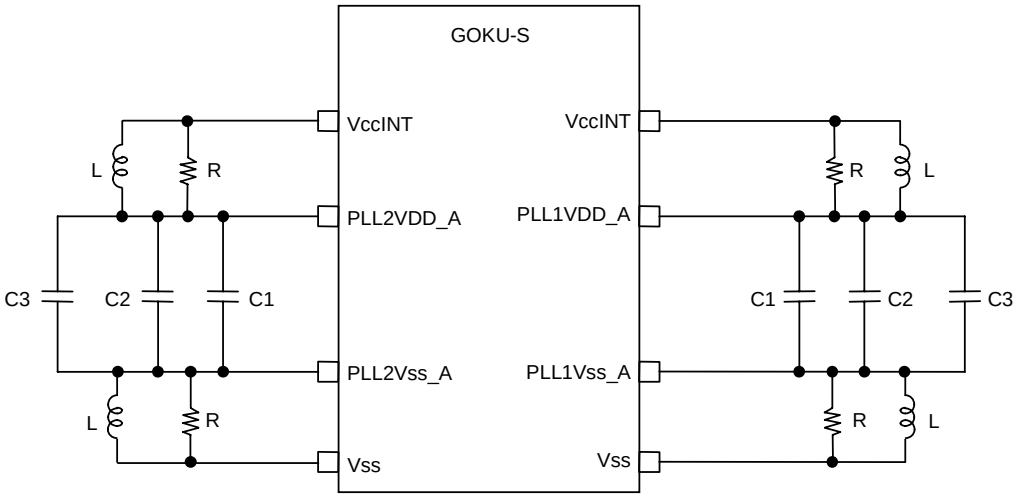
(3) Please surround the crystal area with GND.

(Ta = 0 ~ 85 °C, V_{CCIO} = 3.3 V ± 0.2 V, V_{CCInt} = 1.5 V ± 0.1 V, V_{SS} = 0 V)

Parameter	Symbol	Recommended Value		Unit
		Min	Max	
Crystal Oscillator Frequency	f_{IN}	7.9996	8.0004	MHz
External Capacitors	C_{IN}, C_{OUT}	10	33	pF

12.5 Power Circuit for PLL

12.5.1 Recommended Circuit for PLL



Note: C1, C2, C3, R and L should be placed as closed to the GOKU-S as possible.

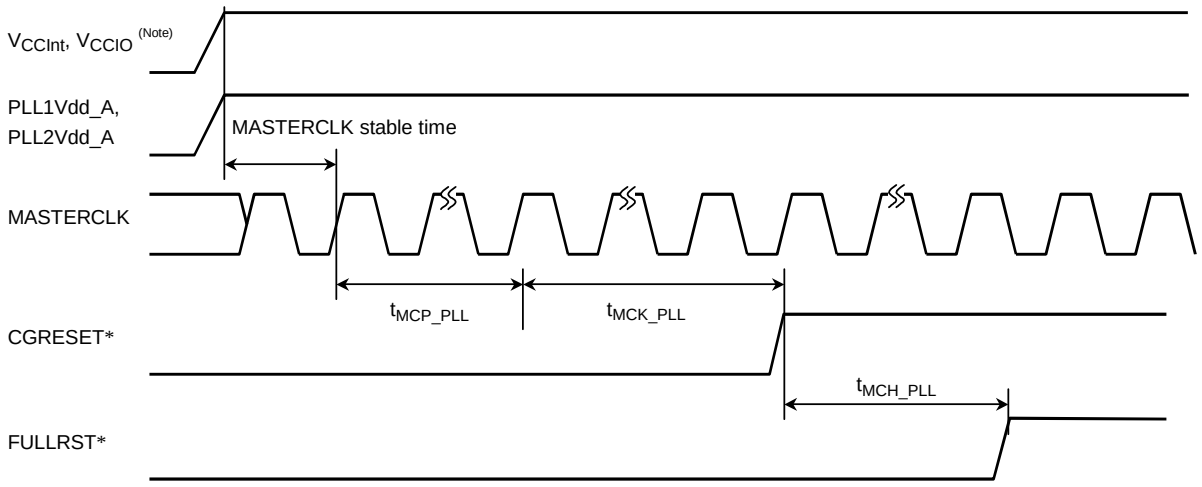
Parameter	Symbol	As a Reference Value	Unit
Resistor	R	5	Ohm
Inductance	L	T.B.D.	μH
Capacitor	C1	1	nF
	C2	82	nF
	C3	10	μF
VCCInt, PLL1VDD_A, PLL2VDD_A		1.5 V ± 0.1 V	V

12.6 AC Characteristics

12.6.1 Power on AC Characteristics

(Ta = 0 ~ 85 °C, V_{CCIO} = 3.3 V ± 0.2 V, V_{CCInt} = 1.5 V ± 0.1 V, V_{SS} = 0 V)

Parameter	Symbol	Condition	Min	Max	Unit
PLL Stable Time	tMCP_PLL		1		ms
CGRESET* Width Time	tMCK_PLL		1		ms
FULLRST* Width Time	tMCH_PLL		1	—	ms



(Note) V_{CCInt} and V_{CCIO} must start up simultaneously, or V_{CCInt} must be first.
The difference of the stand up time of a power supply within in 100 m seconds.

12.6.2 AC Characteristics of PCI 3.3 V Interface

(Ta = 0 ~ 85 °C, V_{CCIO} = 3.3 V ± 0.2 V, V_{CCInt} =1.5 V ± 0.1 V, V_{SS} = V)

Parameter	Symbol	Condition	Min	Max	Unit
PCICLKIN Cycle Time	t_{CYC}		30	40	Ns
PCICLKIN High Time	t_{HIGH}		11		Ns
PCICLKIN Low Time	t_{LOW}		11		Ns
PCICLKIN Slew Rate	t_{SLEW}		1	4	V/ns
PCI Output Signal (1) Output Delay	t_{VAL}	C_L = 70 pF	2	11	Ns
PCI Output Signal (2) Input Setup Time	t_{SU}		7	—	Ns
PCI Input Signal (2) Input Hold Time	t_{HO}		0		Ns
IDSEL, REQ*, GNT* Output Delay	t_{VALPP}	C_L = 70 pF, point to point connect	2	12	Ns
IDSEL, REQ*, GNT* Input Setup Time	t_{SUPP}	point to point connect	10	—	Ns
IDSEL, REQ*, GNT* Input Hold Time	t_{HOPP}	point to point connect	0		Ns

(1) AD[31:0], C_BE[3:0], PAR, FRAME*, IRDY*, TRDY*, STOP*, DEVSEL*, PERR*, SERR*,

(2) AD[31:0], C_BE[3:0], PAR, FRAME*, IRDY*, TRDY*, STOP*, DEVSEL*, PERR*, SERR*, IDSEL

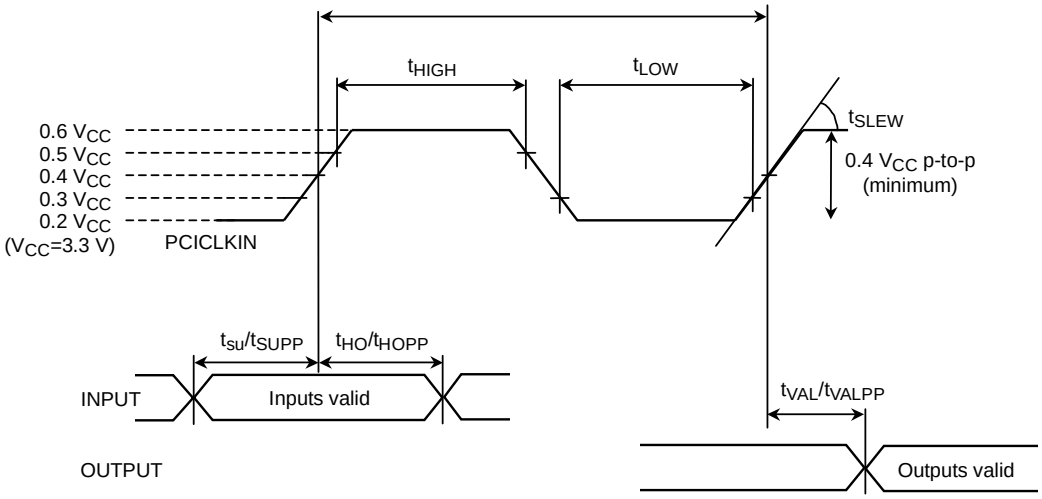


Figure 12.6.1 PCI Interface (3.3 V)

12.6.3 AC Characteristics of PCI EEPROM Interface

(Ta = 0 ~ 85 °C, V_{CCIO} = 3.3 V ± 0.2 V, V_{CCInt} = 1.5 V ± 0.1 V, V_{SS} = 0 V)

Parameter	Symbol	Condition	Min	Max	Unit
EEPROM_SK High Time	t_{HIGH_EPSK}	C_L=50 pF	500		Ns
EEPROM_SK Low Time	t_{LOW_EPSK}	C_L=50 pF	500		Ns
EEPROM_DO Output Delay Time(1)	t_{VAL_EPDO}	C_L=50 pF		100	Ns
EEPROM_DI Input Setup Time	t_{SU_EPDI}		100		Ns
EEPROM_DI Input Hold Time	t_{HO_EPDI}		100		Ns
EEPROM_CS Output Delay Time	t_{VAL_CS}	C_L=50 pF	100		Ns

(1) The GOKU-S controls the EEPROM control signals in synchronization with the falling edge of EEPROM_SK. Because the EEPROM operates with the rising edge of EEPROM_SK, EEPROM_DO has no minimum timing constraint.

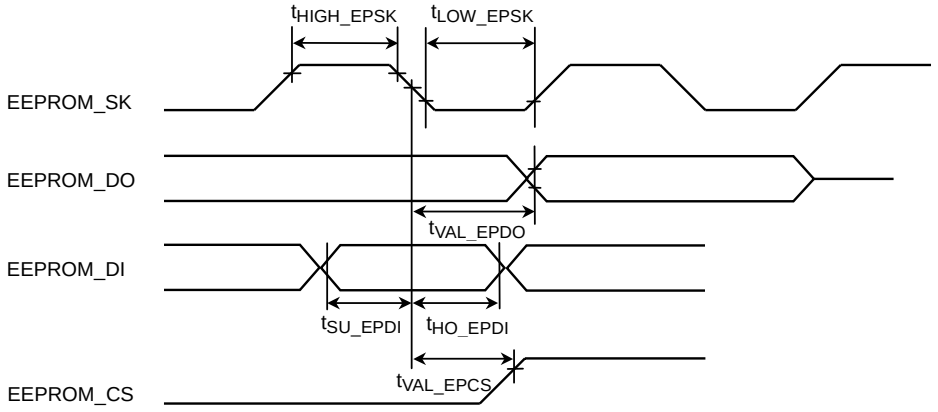


Figure 12.6.2 PCI EEPROM Interface

12.6.4 AC Characteristics of ATA Interface

Table 12.6.1 AC Characteristics of ATA Interface

(Ta = 0 ~ 85 °C, V_{CCIO} = 3.3 V ± 0.2 V, V_{CCInt} = 1.5 V ± 0.1 V, V_{SS} = 0 V)

Description		Min	Max
SRISE	Rising edge slew rate for any signal on AT interface (see note)		1.25 V/ns
SFALL	Falling edge slew rate for any signal on AT interface (see note)		1.25 V/ns
Chost	Host interface signal capacitance at the host connector		25 pf
Cdevice	Device interface signal capacitance at the device connector		20 pf
Note – SRISE and SFALL shall meet this requirement when measured at the sender's connector from 10-90% of full signal amplitude with all capacitive loads from 15 pf through 40 pf where all signals have the same capacitive load value.			

12.6.5 AC Characteristics of USB Host/Device

Table 12.6.2 AC Characteristics of USB Full-speed Source AC Characteristics (for USB Host/USB Device)

(Ta = 0 ~ 85 °C, V_{CCIO} = 3.3 V ± 0.2 V, V_{CCInt} = 1.5 V ± 0.1 V, V_{SS} = 0 V)

Parameter	Symbol	Conditions	Min	Max	Unit
Driver Characteristics:					
Rise Time	T_{FR}	CL = 50 pF	4	20	ns
Fall Time	T_{FF}	CL = 50 pF	4	20	ns
Differential Rise and Fall Time Matching	T_{FRFM}	(T_{FR}/T_{FF})	90	110	%
Driver Output Resistance	Z_{DRV}	Steady state drive	28	44	Ω
Clock Timings:					
Full-speed Data Rate	T_{FDRATE}	Average bit rate = 12 Mb/s ± 0.25 %	11.9700	12.0300	Mb/s
Frame Interval	T_{FRAME}		0.9995	1.0005	ms
Full-speed Data Timings:					
Source Jitter Total (including frequency tolerance):		Note 1, 2			
To Next Transition	T_{DJ1}		−3.5	3.5	ns
For Paired Transitions	T_{DJ2}		−4	4	ns
Source Jitter for Differential Transition to SE0 Transition	T_{FDEOP}	Note 2	−2	5	ns
Receiver Jitter:		Note 2			
To Next Transition	T_{JR1}		−18.5	18.5	ns
For Paired Transitions	T_{JR2}		−9	9	ns
Source SE0 interval of EOP	T_{FEOPT}		160	175	ns
Receiver SE0 interval of EOP	T_{FEOPR}		TBD		ns

Table 12.6.3 AC Characteristics of USB Low-Speed Source (USB Host Only)

(Ta = 0 ~ 85 °C, V_{CCIO} = 3.3 V ± 0.2 V, V_{CCInt} = 1.5 V ± 0.1 V, V_{SS} = 0 V)

Parameter	Symbol	Condition	Min	Max	Unit
Driver Characteristics:					
Transition Time: Rise Time Fall Time	 T_{LR} T_{LF}	Min measured with :CL = 50 pF Max measured with :CL = 350 pF	 75 75	 300 300	 ns ns
Rise and Fall Time Matching	T_{LRFM}	(T_{LR}/T_{LF})	80	120	%
Clock Timings:					
Low-speed Data Rate	T_{LDRATE}	Average bit rate = 1.5 Mb/s ±1.5 %	1.4775	1.5225	Mb/s
Low-speed Data Timings:					
Upstream Port Source Jitter Total (Including Frequency Tolerance): To Next Transition For Paired Transitions	 T_{UDJ1} T_{UDJ2}	Note 1, 2	 −95 −150	 95 150	 ns ns
Upstream Port Source Jitter for Differential Transition to SE0 Transition	T_{LDEOP}	Note 2	−40	100	ns
Upstream Port Differential Receiver Jitter: To Next Transition For Paired Transitions	 T_{DJR1} T_{DJR2}	Note 2	 −75 −45	 75 45	 ns ns
Downstream Port Source Jitter Total (including frequency tolerance): To Next Transition For Paired Transitions	 T_{DDJ1} T_{DDJ2}	Note 1, 2	 TBD TBD	 TBD TBD	 ns ns
Downstream port Differential Receiver Jitter: To Next Transition For Paired Transitions	 T_{UJR1} T_{UJR2}		 −152 −200	 152 200	 ns ns
Source SE0 Interval of EOP	T_{LEOPT}		1.25	1.50	µs
Receiver SE0 Interval of EOP	T_{LEOPR}		TBD		ns

Note 1: Timing difference between the differential data signals.

Note 2: Measured at crossover point of differential data signals.

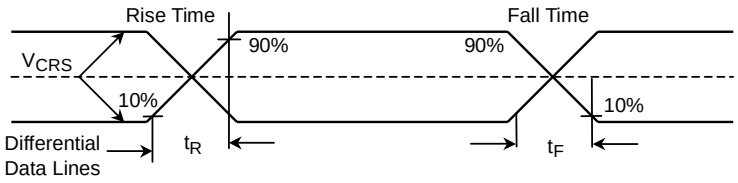


Figure 12.6.3 Data Signal Rise and Fall Time

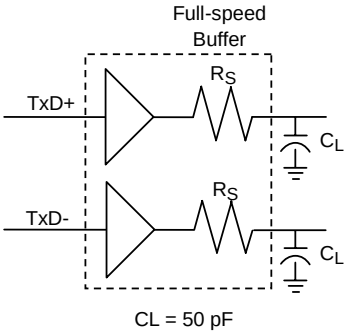


Figure 12.6.4 Full-speed Load

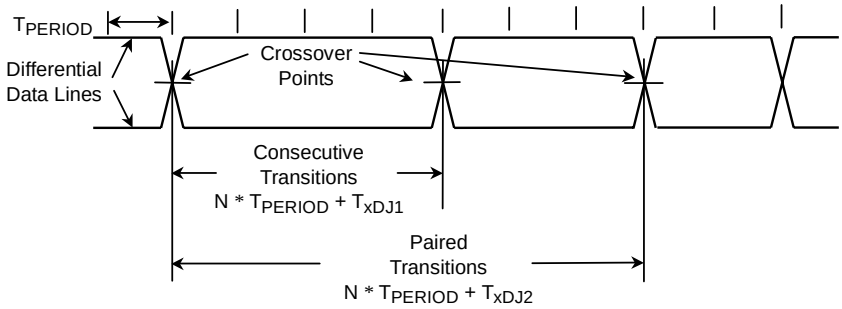


Figure 12.6.5 Differential Data Jitter

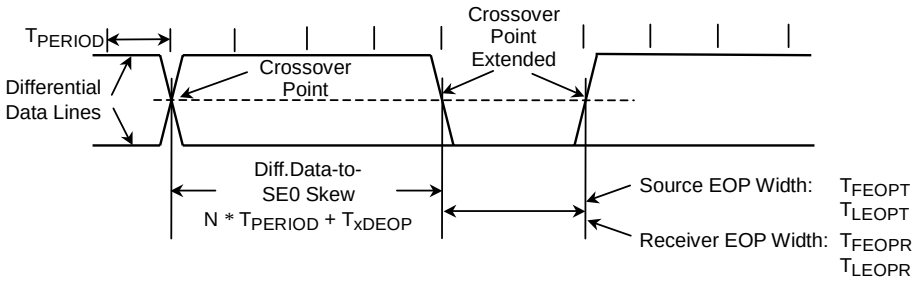


Figure 12.6.6 Differential - to - EOP Transition Skew and EOP Width

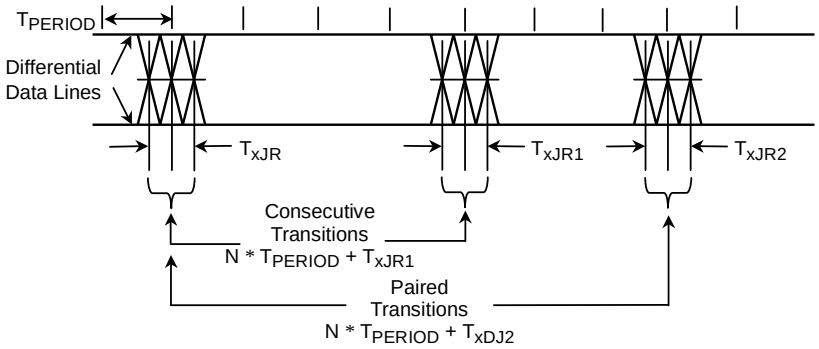


Figure 12.6.7 Receiver Jitter Tolerance

12.6.6 AC Characteristics of I^2C Interface Pins

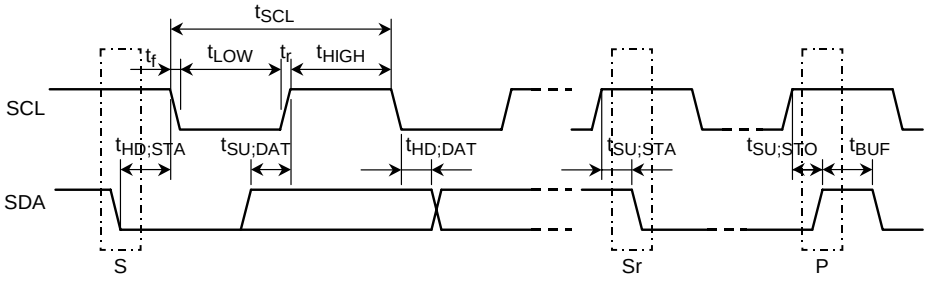
Table 12.6.4 AC Characteristics of I^2C Interface Pins

($Ta = 0 \sim 85$ °C, $V_{CCIO} = 3.3\ V \pm 0.2\ V$, $V_{CCInt} = 1.5\ V \pm 0.1\ V$, $V_{SS} = 0\ V$)

Symbol	Parameter	Standard-Mode		Fast-Mode		Unit
		Min	Max	Min	Max	
fSCL	SCL clock frequency	0	62.7	0	396.8	kHz
tHD;STA	Hold time (repeated) START condition. After this period, the first clock pulse is generated	4.0	—	0.6	—	μs
tLOW	LOW period of the SCL clock	4.7	—	1.3	—	μs
tHIGH	HIGH period of the SCL clock	4.0	—	0.6	—	μs
tSU;STA	Set-up time for a repeated START condition	4.7	—	0.6	—	μs
tHD;DAT	Data hold time:	0 (1)		0 (1)		μs
tSU;DAT	Data set-up time	250		100 (2)		μs
tSU;STO	Set-up time for STOP condition	4.0		0.6		μs
tBUF	Bus free time between a STOP and START condition	4.7		1.3		μs

Notes

1. In the I^2C-bus Specification Version 2.1 from Philips Semiconductors, a device must internally provide a hold time of at least 300 ns for the SDA signal (referred to the VIHmin of the SCL signal) to bridge the undefined region of the falling edge of SCL. But this device is not satisfied in this condition. And this device has not the output buffers incorporate slope control of the falling edges of the SDA and SCL signals. Please keep Data hold time which include tr/tf of SDA/SCL of Table 12.6.4 above.
2. A Fast-mode I2C-bus device can be used in a Standard-mode I^2C-bus system, but the requirement $tSU;DAT \geq 250$ ns must then be met. This will automatically be the case if the device does not stretch the LOW period of the SCL signal. If such a device does stretch the LOW period of the SCL signal, it must output the next data bit to the SDA line tr max + tSU;DAT = 1000 + 250 = 1250 ns (according to the Standard-mode I^2C-bus specification) before the SCL line is released.
3. Although the I^2C-bus Specification Version 2.1 from Philips Semiconductors that I/O pins of Fast-mode devices must not obstruct the SDA and SCL lines if VDD is switched off, the TC86C001 does not comply with this requirement.



S: Start Condition
Sr: Re-startCondition
P; Stop Condition

12.6.7 SIO Interface AC Characteristics

(Ta = 0 ~ 85 °C, V_{CCIO} = 3.3 V ± 0.2 V, V_{CCInt} = 1.5 V ± 0.1 V, V_{SS} = 0 V)

Parameter	Symbol	Rating	Min	Max	Unit
SCLK Cycle Time	f_{CYC_SCLK}		30 × 1.1	—	ns
SCLK Frequency	f_{SCLK}		—	66.6 × 0.45	MHz
SCLK High Time	t_{HIGH_SCLK}		1/2 × 30 × 1.1	—	ns
SCLK Low Time	t_{LOW_SCLK}		1/2 × 30 × 1.1	—	ns

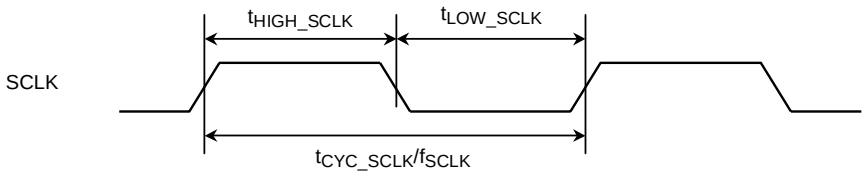


Figure 12.6.8 Timing Diagrams: SIO Interface

12.6.8 PIO Interface AC Characteristics

(Ta = 0 ~ 85 °C, V_{CCIO} = 3.3 V ± 0.2 V, V_{CCInt} = 1.5 V ± 0.1 V, V_{SS} = 0 V)

Parameter	Symbol	Rating	Min	Max	Unit
GPIO[4:0] Output Delay Time	t_{VAL_PIO}	RI2CCLK Standard (C_L=50 pF)	—	13	ns
GPIO[4:0] Input Setup Time	t_{SU_PIO}	RI2CCLK Standard	8.5	—	ns
GPIO[4:0] Input Hold Time	t_{HO_PIO}	R!2CCLK Standard	0	—	ns

Note 1: RI2CCLK is internal clock Please refer 5.1 Clock.

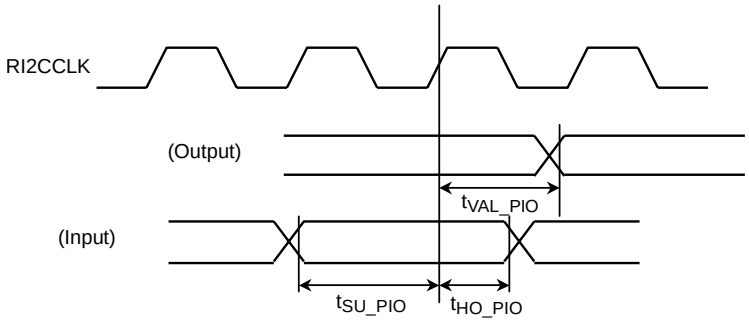


Figure 12.6.9 Timing Diagrams: PIO Interface

12.6.9 ATA Timing

12.6.9.1 Register Transfers

Table 12.6.5 Register Transfer to/from Device

Register Transfer Timing Parameters			Mode 0 ns	Mode 1 ns	Mode 2 ns	Mode 3 ns	Mode 4 ns	Note
t_0	Cycle Time	(min)	600	383	330	180	120	1, 4, 5
t_1	Address Valid to DIOR-/DIOW- Setup	(min)	70	50	30	30	25	
t_2	DIOR-/DIOW- Pulse Width 8-bit	(min)	290	290	290	80	70	1
t_{2i}	DIOR-/DIOW- Recovery Time	(min)	—	—	—	70	25	1
t_3	DIOW- Data Setup	(min)	60	45	30	30	20	
t_4	DIOW- Data Hold	(min)	30	20	15	10	10	
t_5	DIOR- Data Setup	(min)	50	35	20	20	20	
t_6	DIOR- Data Hold	(min)	5	5	5	5	5	
t_{6z}	DIOR- Data Tristate	(max)	30	30	30	30	30	2
t_9	DIOR-/DIOW- to Address Valid Hold	(min)	20	15	10	10	10	
t_{RD}	Read Data Valid to IORDY Active (if IORDY initially low after tA)	(min)	0	0	0	0	0	
t_A	IORDY Setup Time		35	35	35	35	35	3
t_B	IORDY Pulse Width	(max)	1250	1250	1250	1250	1250	
t_C	IORDY Pulse Width	(max)	5	5	5	5	5	

Notes

1 t_0 is the minimum total cycle time, t_2 is the minimum DIOR-/DIOW- assertion time, and t_{2i} is the minimum DIOR-/DIOW- negation time. A host implementation shall lengthen t_2 and/or t_{2i} to ensure that t_0 is equal to or greater than the value reported in the devices IDENTIFY DEVICE data. A device implementation shall support any legal host implementation.

2 This parameter specifies the time from the negation edge of DIOR- to the time that the data bus is released by the device.

3 The delay from the activation of DIOR- or DIOW- until the state of IORDY is first sampled. If IORDY is inactive then the host shall wait until IORDY is active before the register transfer cycle is completed. If the device is not driving IORDY negated at the t_A after the activation of DIOR- or DIOW-, then t_5 shall be met and t_{RD} is not applicable. If the device is driving IORDY negated at the time t_A after the activation of DIOR- or DIOW-, then t_{RD} shall be met and t_5 is not applicable.

4 ATA/ATAPI standards prior to ATA/ATAPI-5 inadvertently specified an incorrect value for mode 2 time t_0 by utilizing the 16-bit PIO value.

5 Mode shall be selected no faster than the highest mode supported by the slowest device.

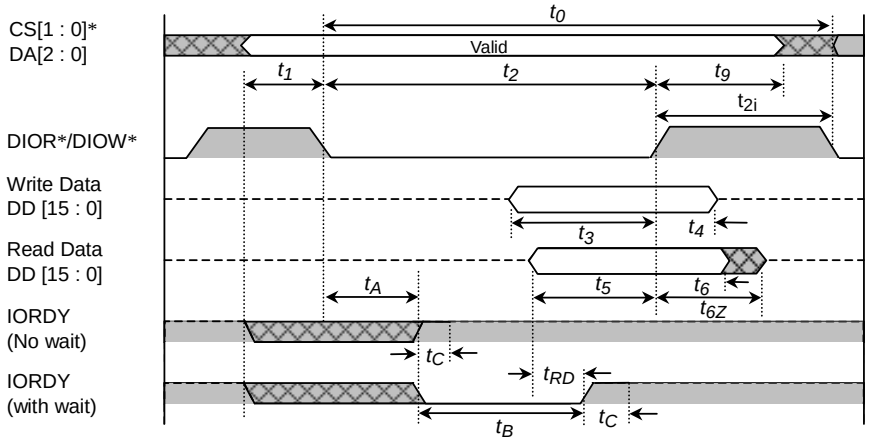


Figure 12.6.10 Register Transfer to/from Device

12.6.9.2 PIO Data Transfers

Table 12.6.6 PIO Data Transfer to/from Device

PIO Timing Parameters			Mode 0 ns	Mode 1 ns	Mode 2 ns	Mode 3 ns	Mode 4 ns	Note
t_0	Cycle Time	(min)	600	383	240	180	120	1, 4
t_1	Address Valid to DIOR-/DIOW- Setup	(min)	70	50	30	30	25	
t_2	DIOR-/DIOW- Pulse Width	(min)	165	125	100	80	70	1
t_{2i}	DIOR-/DIOW- Recovery Time	(min)	—	—	—	70	25	1
t_3	DIOW- Data Setup	(min)	60	45	30	30	20	
t_4	DIOW- Data Hold	(min)	30	20	15	10	10	
t_5	DIOR- Data Setup	(min)	50	35	20	20	20	
t_6	DIOR- Data Hold	(min)	5	5	5	5	5	
t_{6z}	DIOR- Data Tristate	(max)	30	30	30	30	30	2
t_9	DIOR-/DIOW- to Address Valid Hold	(min)	20	15	10	10	10	
t_{RD}	Read Data Valid to IORDY Active (if IORDY initially low after tA)	(min)	0	0	0	0	0	
t_A	IORDY Setup Time		35	35	35	35	35	3
t_B	IORDY Pulse Width	(max)	1250	1250	1250	1250	1250	
t_C	IORDY Pulse Width	(max)	5	5	5	5	5	

Notes

1 t_0 is the minimum total cycle time, t_2 is the minimum DIOR-/DIOW- assertion time, and t_{2i} is the minimum DIOR-/DIOW- negation time. A host implementation shall lengthen t_2 and/or t_{2i} to ensure that t_0 is equal to or greater than the value reported in the devices IDENTIFY DEVICE data. A device implementation shall support any legal host implementation.

2 This parameter specifies the time from the negation edge of DIOR- to the time that the data bus is released by the device.

3 The delay from the activation of DIOR- or DIOW- until the state of IORDY is first sampled. If IORDY is inactive then the host shall wait until IORDY is active before the PIO cycle is completed. If the device is not driving IORDY negated at the t_A after the activation of DIOR- or DIOW-, then t_5 shall be met and t_{RD} is not applicable. If the device is driving IORDY negated at the time t_A after the activation of DIOR- or DIOW-, then t_{RD} shall be met and t_5 is not applicable.

4 Mode may be selected at the highest mode for the device if CS(1:0) and AD(2:0) do not change between read or write cycles or selected at the highest mode supported by the slowest device if CS(1:0) or AD(2:0) do change between read or write cycles.

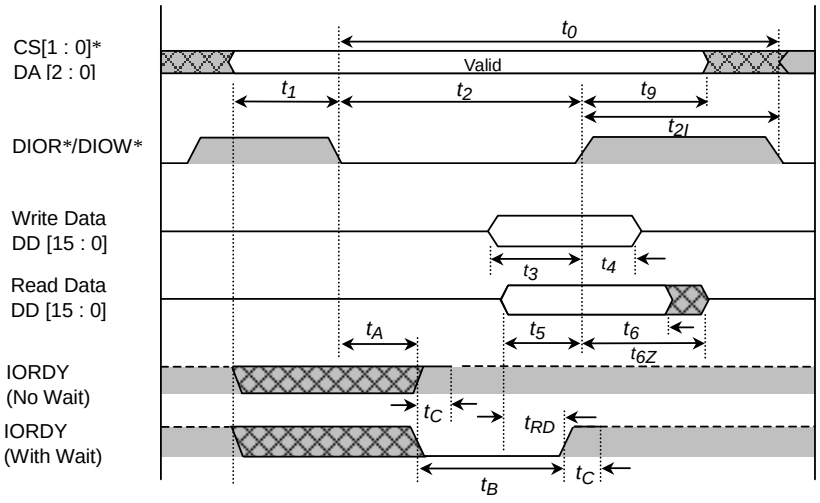


Figure 12.6.11 PIO Data Transfer to/from Device

12.6.9.3 Multiword DMA Data Transfer

Table 12.6.7 Multiword DMA Data Transfer

Multiword DMA Timing Parameters			Mode 0 ns	Mode 1 ns	Mode 2 ns	Note
t0	Cycle Time	(min)	480	150	120	see note
tD	DIOR-/DIOW- Asserted Pulse Width	(min)	215	80	70	see note
tE	DIOR- Data Access	(max)	150	60	50	
tF	DIOR- Data Hold	(min)	5	5	5	
tG	DIOR-/DIOW- Data Setup	(min)	100	30	20	
tH	DIOW- Data Hold	(min)	20	15	10	
ti	DMACK to DIOR-/DIOW- Setup	(min)	0	0	0	
tj	DIOR-/DIOW- to DMACK Hold	(min)	20	5	5	
tKr	DIOR- Negated Pulse Width	(min)	50	50	25	see note
tKw	DIOW- Negated Pulse Width	(min)	215	50	25	see note
tLr	DIOR- to DMARQ Delay	(max)	120	40	35	
tLw	DIOW- to DMARQ Delay	(max)	40	40	35	
tM	CS(1:0) Valid to DIOR-/DIOW-	(min)	50	30	25	
tN	CS(1:0) Hold	(min)	15	10	10	
tZ	DMACK- to Read Data Released	(max)	20	25	25	

Note: t0 is the minimum total cycle time, tD is the minimum DIOR-/DIOW- assertion time, and tK (tKR or tKW, as appropriate) is the minimum DIOR-/DIOW- negation time. A host shall lengthen tD and/or tK to ensure that t0 is equal to the value reported in the devices IDENTIFY DEVICE data.

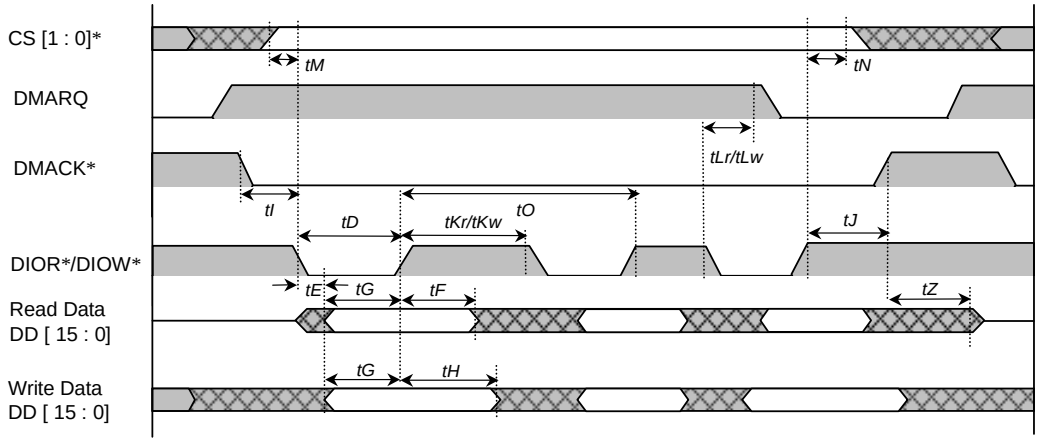


Figure 12.6.12 Multiword DMA Data Transfer to/from Device

12.6.9.4 Ultra DMA Data Transfers

Table 12.6.8 Ultra DMA Data Burst Timing Requirements

Name	Mode 0 (in ns)		Mode 1 (in ns)		Mode 2 (in ns)		Mode 3 (in ns)		Mode 4 (in ns)		Comment (See Note 1 and 2)
	Min	Max	Min	Max	Min	Max	Min	Max	Min	Max	
$t_{2CYCTYP}$	240		160		120		90		60		Typical sustained average two cycle time
t_{CYC}	112		73		54		39		25		Cycle time allowing for asymmetry and clock variations (from STROBE edge to STROBE edge)
t_{2CYC}	230		154		115		86		57		Two cycle time allowing for clock variations (from rising edge to next rising edge or from falling edge to next falling edge of STROBE)
t_{DS}	15		10		7		7		5		Data setup time at recipient
t_{DH}	5		5		5		5		5		Data hold time at recipient
t_{DVS}	70		48		30		20		6		Data valid setup time at sender (from data valid until STROBE edge) (see Note 4)
t_{DVH}	6		6		6		6		6		Data valid hold time at sender (from STROBE edge until data may become invalid) (see Note 4)
t_{FS}	0	230	0	200	0	170	0	130	0	120	First STROBE time (for device to first negate DSTROBE from STOP during a data in burst)
t_{LI}	0	150	0	150	0	150	0	100	0	100	Limited interlock time (see Note 3)
t_{MLI}	20		20		20		20		20		Interlock time with minimum (see Note 3)
t_{UI}	0		0		0		0		0		Unlimited interlock time (see Note 3)
t_{AZ}		10		10		10		10		10	Maximum time allowed for output drivers to release (from asserted or negated)
t_{ZAH}	20		20		20		20		20		Minimum delay time required for output
t_{ZAD}	0		0		0		0		0		drivers to assert or negate (from released)
t_{ENV}	20	70	20	70	20	70	20	55	20	55	Envelope time (from DMACK- to STOP and HDMARDY- during data in burst initiation and from DMACK to STOP during data out burst initiation)
t_{SR}		50		30		20		NA		NA	STROBE-to-DMARDY- time (if DMARDY- is negated before this long after STROBE edge, the recipient shall receive no more than one additional data word)
t_{RFS}		75		70		60		60		60	Ready-to-final-STROBE time (no STROBE edges shall be sent this long after negation of DMARDY-)
t_{RP}	160		125		100		100		100		Minimum time to assert STOP or negate DMARQ
t_{IORDYZ}		20		20		20		20		20	Maximum time before releasing IORDY
t_{ZIORDY}	0		0		0		0		0		Minimum time before driving STROBE (see note 5)
t_{ACK}	20		20		20		20		20		Setup and hold times for DMACK- (before assertion or negation)
t_{SS}	50		50		50		50		50		Time from STROBE edge to negation of DMARQ or assertion of STOP (when sender terminates a burst)

Notes

1 Timing parameters shall be measured at the connector of the sender or receiver to which the parameter applies. For example, the sender shall stop generating STROBE edges t_{RFS} after the negation of DMARDY-. Both STROBE and DMARDY- timing measurements are taken at the connector of the sender.

2 All timing measurement switching points (low to high and high to low) shall be taken at 1.5 V.

3 t_{UI}, t_{MLI}, and t_{LI} indicate sender-to-recipient or recipient-to-sender interlocks, i.e., either sender or recipient is waiting for the other to respond with a signal before proceeding. t_{UI} is an unlimited interlock that has no maximum time value. t_{MLI} is a limited time-out that has a defined minimum. t_{LI} is a limited time-out that has a defined maximum.

4 The test load for t_{DVS} and t_{DVH} shall be a lumped capacitor load with no cable or receivers. Timing for t_{DVS} and t_{DVH} shall be met for all capacitive loads from 15 to 40 pf where all signals have the same capacitive load value.

5 t_{ZIORDY} may be greater than t_{ENV} since the device has a pull up on IORDY- giving it a known state when released.

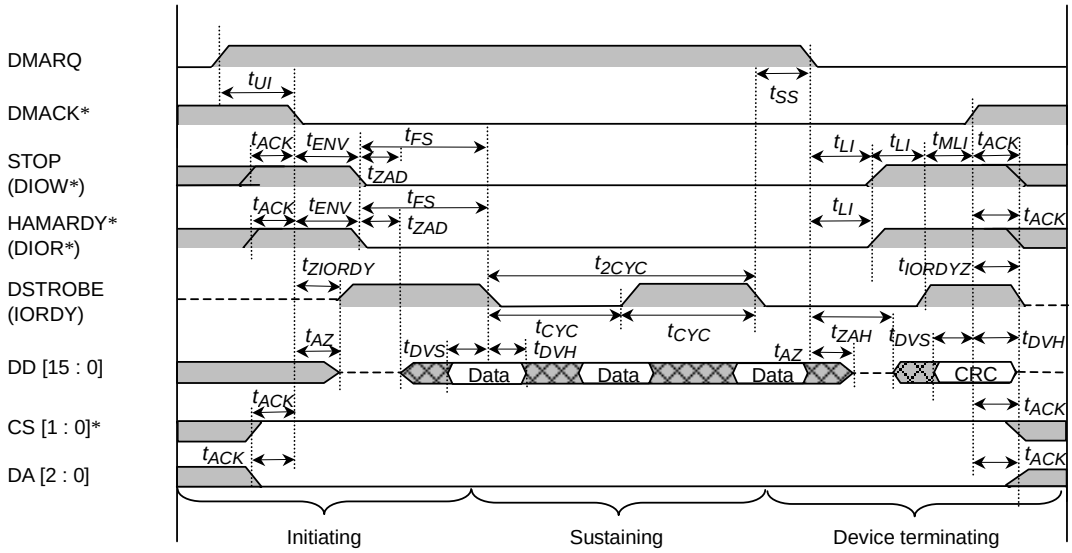


Figure 12.6.13 Ultra DMA data Transfer to/from device

13. Package and Pin Assignment

13.1 Pin Assignment

Figure 13.1.1 Pin Designations of TC86C001FG (GOKU-S)

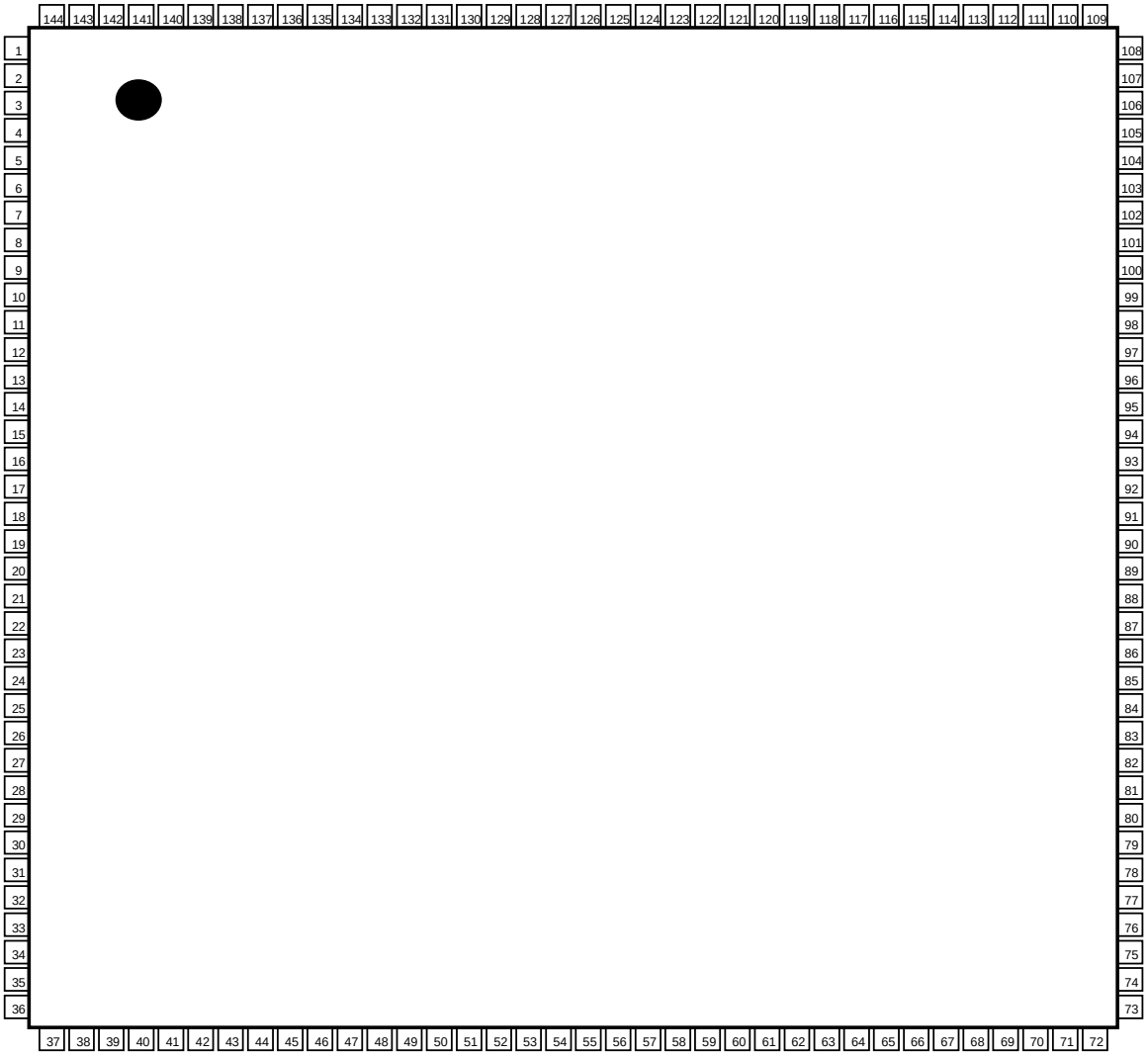


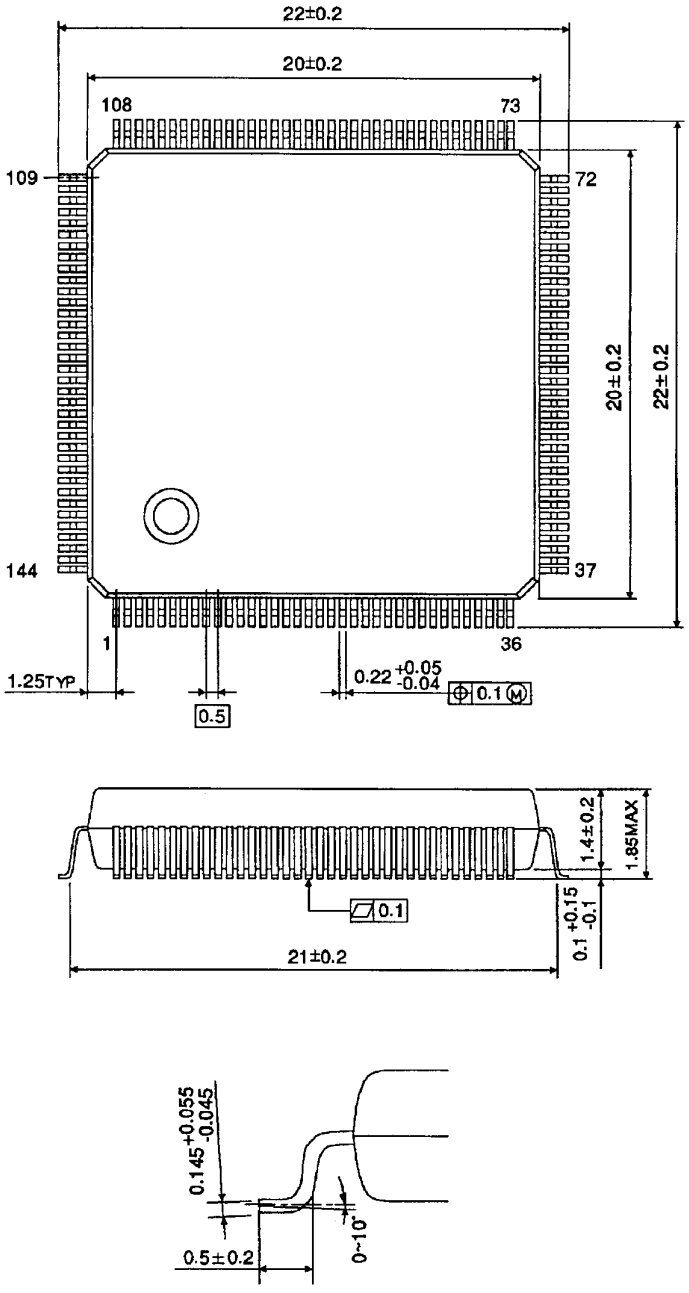
Table 13.1 Pin Designations (LQFP144)

Pin No.	Pin Function	Pin No.	Pin Function	Pin No.	Pin Function	Pin No.	Pin Function
1	V_{CCIO}	37	AD9	73	SCLK/GPIO0	109	DD02
2	AD24	38	AD8	74	V_{CCIO}	110	DD12
3	C/BE3	39	V_{CCIO}	75	TRST*	111	DD03
4	IDSEL	40	V_{SS}	76	MODE1*	112	V_{SS}
5	AD23	41	C/BE0	77	V_{SS}	113	DD11
6	AD22	42	AD7	78	FULLRST*	114	DD04
7	AD21	43	AD6	79	V_{CCINT}	115	DD10
8	V_{SS}	44	AD5	80	MODE0*	116	DD05
9	AD20	45	AD4	81	CGRESET*	117	DD09
10	AD19	46	AD3	82	V_{SS}	118	DD06
11	AD18	47	V_{SS}	83	USBHDP1	119	DD08
12	AD17	48	AD2	84	USBHDN1	120	DD07
13	AD16	49	AD1	85	V_{CCIO}	121	RESET*
14	C/BE2	50	AD0	86	EEPROM_CS/TDO	122	V_{CCIO}
15	V_{SS}	51	V_{CCIO}	87	EEPROM_DO	123	USBDDP
16	FRAME*	52	USBHDP0	88	CS1*	124	USBDDN
17	IRDY*	53	USBHDN0	89	CS0*	125	USBDPULLUP
18	V_{CCINT}	54	V_{SS}	90	V_{CCIO}	126	USBDPDECT
19	V_{CCIO}	55	USBHPON0	91	DA2	127	V_{SS}
20	TRDY*	56	USBHOC0*/TMS	92	DA0/EEPROM_DI	128	INTA*
21	DEVSEL*	57	USBHPON1	93	V_{SS}	129	INTB*
22	V_{SS}	58	USBHOC1*/TDI	94	PDIAG*	130	V_{CCINT}
23	STOP*	59	V_{CCIO}	95	DA1/EEPROM_SK	131	PCICLK
24	V_{SS}	60	TCK	96	INTRQ	132	V_{SS}
25	PERR*	61	V_{CCINT}	97	DMACK*	133	GNT*
26	SERR*	62	XIN	98	IORDY	134	REQ*
27	PAR	63	XOUT	99	DIOR*	135	AD31
28	C/BE1	64	V_{SS}	100	DIOW*	136	V_{SS}
29	V_{CCIO}	65	PLL1VSS_A	101	DMARQ	137	AD30
30	AD15	66	PLL1VDD_A	102	V_{SS}	138	AD29
31	AD14	67	PLL2VDD_A	103	DD15	139	AD28
32	V_{SS}	68	PLL2VSS_A	104	DD00	140	V_{CCIO}
33	AD13	69	RTS*/SDA/GPIO4	105	DD14	141	AD27
34	AD12	70	CTS*/SCL/GPIO3	106	DD01	142	AD26
35	AD11	71	TXD/GPIO2	107	DD13	143	AD25
36	AD10	72	RXD/GPIO1	108	V_{CCIO}	144	V_{SS}

13.2 Package Dimensions

144 Lead LQFP (LQFP144-P-2020-0.50A)

Unit : mm



14. TC86C001FG Usage Limitations

14.1 Limitation on SIO Break Detection

[Limitation]

If an all-0 character has been received without a preceding start bit, it is not recognized as a break. The receive operation terminates only with a framing error indication.

[Workaround]

When sending a break to the GOKU-S, precede an all-0 character with a start bit.

14.2 Limitation #1 on the ATA/ATAPI DMA Transfer Size

[Limitation]

As specified in the “Programming Interface for Bus Master IDE Controller Revision 1.0,” the total sum of the Physical Region Descriptor (PRD) byte counts must be equal to the size of the ATA/ATAPI device transfer request.

[Effects of a violation]

If the size of the device transfer request is greater than (descriptor byte count + 1024) for a DMA read (from the ATA bus to the PCI bus), the transfer might freeze.

(Such programming does not comply with the bus master IDE specification; software error handling is required. Typically, match the size of the device transfer request with the total sum of the descriptor byte counts.)

[Solution]

If the size of the device transfer request is greater on a DMA read transfer, part of the transfer from the ATA/ATAPI device will still remain unfinished at the time when the transfer to the PCI bus has been completed. Consequently, the INTRQ signal is not asserted by the ATA/ATAPI device. In this case, a software timeout must be used to check the DMA status. After a timeout, first write 0x00 to the Command register (at offset 0x00) described in Section 6.2.4.1 of this manual before accessing other ATA registers. Note that because the DMA transfer from the ATA/ATAPI device ended prematurely, the device must be reset before re-initiating a DMA transfer.

14.3 Limitation #2 on the ATA/ATAPI DMA Transfer Size

[Limitation]

As specified in the “Programming Interface for Bus Master IDE Controller Revision 1.0,” the total sum of the Physical Region Descriptor (PRD) byte counts must be equal to the size of the ATA/ATAPI device transfer request.

[Effects of a violation]

If the size of the device transfer request is smaller for a DMA read (from the ATA bus to the PCI bus), a PCI Master Abort might occur.

The GOKU-S is capable of performing posted writes on the PCI bus. Thus, when a DMA read is finished, it generates a read-back cycle on the PCI bus to assure completion of the memory writes. If the size of the device transfer request is smaller, read-back cycles will take place endlessly, incrementing memory addresses.

(Such programming does not comply with the bus master IDE specification; software error handling is required. Typically, match the size of the device transfer request with the total sum of the descriptor byte counts.)

[Solution]

If the size of the device transfer request is smaller on a DMA read transfer, part of the transfer to the PCI bus will still remain unfinished at the time when the transfer from the ATA/ATAPI device has been completed. Consequently, the INTRQ signal is asserted by the ATA/ATAPI device, causing a PCI interrupt immediately after the first read-back cycle. Normally, the software checks the state of Function 0 by accessing the Status register after detecting the interrupt, but in this case, the DMA still remains active. Because PCI read-back cycles are repeated while the DMA is active, the DMA must be terminated immediately.

14.4 Limitation on Small ATAPI Sector Sizes for a DMA Transfer

[Limitation]

A DMA transfer can not be performed with an ATAPI sector size smaller than 512 bytes.

[Effects of a violation]

A DMA transfer is not performed correctly. Correct operation can not be guaranteed.

[Solution]

If the ATAPI sector size is smaller than 512 bytes, use the PIO.

14.5 Limitation on ATA/ATAPI DMA

[Limitation]

In case an ATA/ATAPI DMA transfer is terminated abnormally, a time-out must be detected by the software interrupt handler. On time-out, the interrupt handler must initiate a dummy DMA cycle so that the pending interrupt is serviced.

[Effects of a violation]

If a ATA/ATAPI DMA transfer is terminated abnormally, the Bus Master IDE Controller leaves the interrupt request (INTRQ) from the ATA/ATAPI device pending and does not generate a PCI interrupt (INTA* or INTB*). Consequently, no PCI interrupt will occur until the ATA/ATAPI DMA transfer has successfully been completed.

[Workaround]

On time-out, the software interrupt handler must perform the following sequence of steps to initiate a dummy DMA cycle so that the pending interrupt is generated:

1. Stop the DMA transfer that has been terminated abnormally.

   Reset START_STOPBM (Start/Stop Bus Master) (bit[0]) in the Command register (at offset +0x00).
2. Program the Bus Master IDE Controller as follows:
   a) Write 0x0000 to the Sector Count register (at offset +0x0A).
   b) Write 0x0001 for the Master Device or 0x0002 for the Slave Device to the Sector Count Control register (at offset +0x0C).
   c) Write 0x0000 to the Transfer Word Count 1 register (at offset +0x04) for the Master Device or the Transfer Word Count 2 register (at offset +0x06) for the Slave Device.
3. Initiate a dummy DMA cycle.

   Set START_STOPBM (Start/Stop Bus Master) (bit[0]) in the Command register (at offset +0x00).

   Program R_OR_WCTR (Read or Write Control) (bit[3]) in the Command register for a write control.
4. The pending interrupt (INTA* or INTB*) is generated.
5. Stop the dummy DMA cycle.

   Reset START_STOPBM (Start/Stop Bus Master) (bit[0]) in the Command register (at offset +0x00).

Note: If necessary, save and restore the following registers before and after the dummy DMA cycle:
a) Sector Count register (at offset +0x0C)
b) Sector Count register (at offset +0x0C)
c) Transfer Word Count 1 register (at offset +0x04)
   Transfer Word Count 2 register (at offset +0x06)

14.6 Limitation on ATAPI DMA

[Limitation]

If the transfer sector size is not a multiple of 16 bytes,

- a dummy DMA cycle must be initiated after completion of a DMA read transfer (from the ATA bus to the PCI bus), or
- the ATA registers must not be accessed after the DMA direction bit is programmed for a DMA read (from the ATA bus to the PCI bus).

[Effects of a violation]

If any ATA register is accessed after the DMA direction bit is programmed for a DMA read (from the ATA bus to the PCI bus), a target retry might occur endlessly, causing the DMA to freeze. The DMA does not freeze for the following cases:

- No ATA register is accessed after the DMA direction bit is programmed for a DMA read (from the ATA bus to the PCI bus).
- The remainder of the transfer size (i.e., transfer_sector_size × transfer_sector_count) divided by 512 is equal to or greater than 16.

[Workarounds]

There are two workarounds. The first one is recommended, though.

a) Don't access any ATA register after the DMA direction bit is programmed for a DMA read (from the ATA bus to the PCI bus).

b) Initiate a dummy DMA cycle in the following sequence after completion of a DMA read (from the ATA bus to the PCI bus). (A dummy DMA cycle does not issue a DMA command to the ATAPI device.)

1. Stop the DMA read (from the ATA bus to the PCI bus).

   Reset START_STOPBM (Start/Stop Bus Master) (bit[0]) in the Command register (at offset +0x00).

2. Program the Bus Master IDE Controller as follows:
   a) Write 0x0000 to the Sector Count register (at offset +0x0A).
   b) Write 0x0001 for the Master Device or 0x0002 for the Slave Device to the Sector Count Control register (at offset +0x0C).
   c) Write 0x0000 to the Transfer Word Count 1 register (at offset +0x04) for the Master Device or the Transfer Word Count 2 register (at offset +0x06) for the Slave Device.
3. Initiate a dummy DMA cycle.

   Set START_STOPBM (Start/Stop Bus Master) (bit[0]) in the Command register (at offset +0x00).

   Program R_OR_WCTR (Read or Write Control) (bit[3]) in the Command register for a write control.

4. Stop the dummy DMA cycle.

   Reset START_STOPBM (Start/Stop Bus Master) (bit[0]) in the Command register (at offset +0x00).

Note: If necessary, save and restore the following registers before and after the dummy DMA cycle:

a) Sector Count register (at offset +0x0C)
b) Sector Count Control register (at offset +0x04)
c) Transfer Word Count 1 register (at offset +0x04)
   Transfer Word Count 2 register (at offset +0x06)

14.7 Limitations on Writing to the Command Register of the USB Device Controller

[Limitation]

While the USB Device Controller (UDC) is transmitting a handshake in response to a token from the host, the Command register of the endpoint involved must not be programmed; otherwise, that endpoint will change to a state different from that specified by the command. You must work around this problem by altering the command usage or the timing when the command is issued.

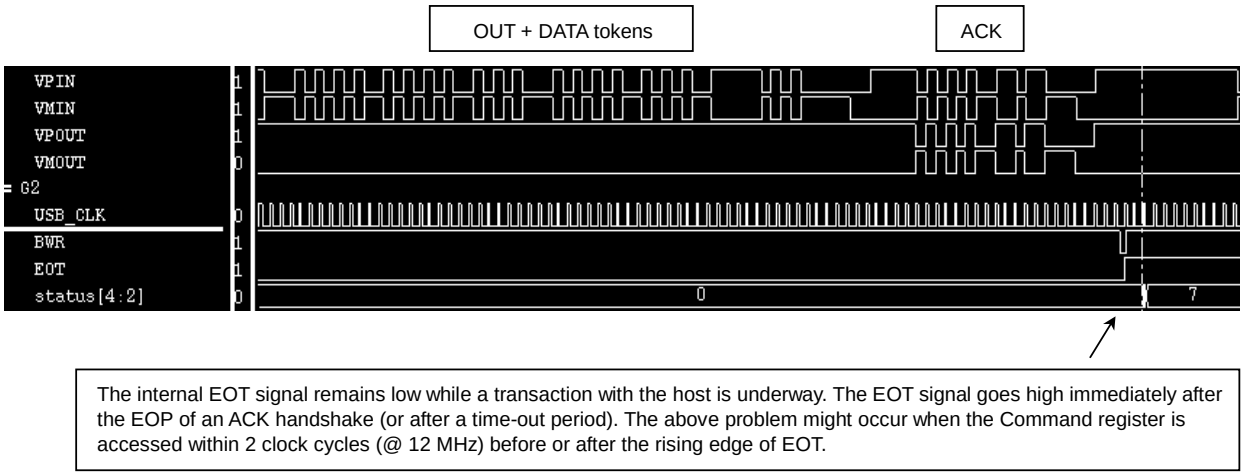


Figure 14.1 Erroneous Behavior

[Effects of a violation]

The effects of a violation differ, depending on the command issued. See the descriptions of Phenomenon 1 and Phenomenon 2 below.

Phenomenon 1: Incorrect transition to the INVALID state on issuance of a STALL command

When the UDC is processing a token for a given endpoint, issuing a STALL command to that endpoint with the timing shown above causes its status to change to INVALID. Once the INVALID state is entered, the UDC becomes non-responsive to any token for that endpoint.

Causes

- Bulk OUT transaction

  Phenomenon 1 only occurs when a STALL command is issued to a bulk OUT endpoint while it is normally receiving data from the host (or transmitting an ACK in response to the received data), as shown in Figure 14.1.

  Phenomenon 1 does not occur if a STALL command is not issued during transmission; a STALL command is issued to a different endpoint; and the endpoint is STALLed in response to a SET_FEATURE request, etc.

- Bulk IN transaction

  Phenomenon 1 can only occur when a STALL command is issued to a bulk IN endpoint while it is transmitting data to the host. If, due to a transmission error, the UDC has not received an ACK handshake from the host for the data it transmitted in response to an IN token (i.e. a time-out occurred), a STALL command causes the UDC to change to the INVALID state. The probability of this happening is considered low.

  Phenomenon 1 does not occur if a STALL command is not issued during transmission; a STALL command is issued to a different endpoint; and the endpoint is STALLed in response to a SET_FEATURE request, etc.

Note: This phenomenon does not occur during control transactions with automatic hardware response. The following problems can only occur regarding control transactions using software response routines.

- Control OUT transaction with a data stage (software response)

  The host accesses the Setup Received register to acknowledge an INT_SETUP interrupt from the UDC. Phenomenon 1 occurs when a STALL command is issued to endpoint 0 (with the same timing as the case of the bulk OUT transaction) while endpoint 0 is receiving data from the host in the data stage.

- Control OUT transaction without a data stage (software response)

  Phenomenon 1 does not occur if the host issues a STALL command between when it has received an INT_SETUP interrupt and when it accesses the EOP register.

- Control IN transaction (software response)

  The host accesses the Setup Received register to acknowledge an INT_SETUP interrupt. The UDC then places data in the endpoint 0 FIFO and sends it out to the host in the data stage. Phenomenon 1 occurs when a transmission error and an issuance of a STALL command happen simultaneously, as is the case with the bulk IN transaction. The probability of this happening is low, as is the case with the bulk IN transaction.

  Phenomenon 1 does not occur if the host issues a STALL command between when it has received an INT_SETUP interrupt and when it transmits data.

Phenomenon 2: Incorrect status indication on issuance of the RESET command

When the UDC is processing a token for a given endpoint, issuing the RESET command to that endpoint causes bits 4:2 in the EPx_STATUS register to show an incorrect status. (The RESET command is executed properly.)

The RESET command should cause the status value to change to 3'b000, but if the RESET command is issued with the timing shown in Figure 14.1, the EPx_STATUS register incorrectly reflects the status that the endpoint was in before the RESET command was issued. The following table shows the status before the RESET command and the resulting status indication.

Status Before RESET	Correct Status Indication that Should Be Set	Incorrect Status Indication Actually Set
DATAIN (3'b001)	READY (3'b000)	DATAIN (3'b001)
FULL (3'b010)	READY (3'b000)	FULL (3'b010)
TX_ERR (3'b011)	READY (3'b000)	TX_ERR (3'b011)
RX_ERR (3'b100)	READY (3'b000)	RX_ERR (3'b100)
STALL (3'b110)	READY (3'b000)	STALL (3'b110)

[Workarounds]

Workarounds for phenomenon 1:

- Bulk IN/OUT transactions

  If an endpoint is performing a bulk transaction, the host must suspend the read or write access to the endpoint, and after receiving a NAK from the UDC, issue a STALL command. Use INT_EPxNAK to acknowledge an NAK.

- Control OUT transactions with a data stage (software response)

  When the host receives INT_SETUP, it must be determined whether endpoint 0 needs to be STALLed to service a request. To STALL endpoint 0, the host must access the Setup Received register, then after acknowledging the INT_ENDPOINT0 interrupt, issue a STALL command.

- Control OUT transactions without a data stage (software response)

  When the host receives INT_SETUP, it must be determined whether endpoint 0 needs to be STALLed to service a request. To STALL endpoint 0, the host must issue a STALL command before accessing the EOP register.

- Control IN transactions (software response)

  When the host receives INT_SETUP, it must be determined whether endpoint 0 needs to be STALLed to service a request. To STALL endpoint 0, the host must issue a STALL command before the UDC places the first transmit data in the FIFO.

Workarounds for phenomenon 2:

If there is any likelihood of the user application exhibiting problematic behaviors due to incorrect status indication, the RESET command must not be issued to an endpoint while it is receiving or transmitting data. (The endpoint must be reset as part of the processing for a request that requires the endpoint to be reset.)

(Typically, endpoints other than EP0 may need to be reset after they are configured or reconfigured to respond to the SET_CONFIG or SET_INTERFACE request by software. Phenomenon 2 does not occur if the RESET command is issued during the processing of a request. The RESET command is also used to reset the specified endpoint to respond to the CLEAR_FEATURE request by software; in this case also, phenomenon 2 does not occur if the RESET command is issued during its processing.

14.8 Limitation on when GOKU-S becomes the Master and Target Write

- Relevant block: Bus Master IDE Controller (Function 0), USB Host Controller (Function 1), USB Device Controller (Function 2)

[Limitation]

When the TC86C001FG (GOKU-S) becomes the Master and a target write is issued, it is no longer possible to guarantee complete writing of target write data to system memory when a GOKU-S interrupt is issued. In order to guarantee write-in completion, GOKU-S writes in and performs 4 bytes of read-out at the last. So GOKU-S can't write directly to the memory which does not correspond to read-out.

[Symptom]

When the TC86C001FG (GOKU-S) becomes the Master and a target write is issued, it is no longer possible to guarantee complete writing of target write data to system memory when a GOKU-S interrupt is issued. In order to guarantee write-in completion, GOKU-S writes in and performs 4 bytes of read-out at the last. However, GOKU-S will repeat read-out and lock when the GOKU-S writes directly to the memory which does not correspond to read-out.

[Workarounds]

1. A DMA transfer is once first performed in the memory in which writing and read-out are possible. Then, a DMA transfer is performed to the address of correspondence of only the target write.
2. In Bus Master IDE Controller and USB Host Controller, it can respond by accepting all data in part to the memory which can be read, without once performing a DMA transfer in the memory in which writing and read-out are possible first, and performing a DMA transfer.

   a) Bus Master IDE Controller

   By adding PRD for 4-byte transfer to a PRD table, it is possible to transmit only 4 bytes of the last to the main memory in which read-out and writing are possible, and to lessen influence on a performance.

   <Example> Case that the HDD data size is 2 sectors (1024 byte):

   Add PRD for 4-byte transfer to a PRD table.

[PRD table]

```
------------1st PRD-------------------------
+ 0x00     (PM_FIFO address)      <- Transfer first 1020 (=1024-4) byte to PM_FIFO
+ 0x04     0x000003FC
------------2nd PRD-------------------------
+ 0x08     (Main Memory Address)  <- Transfer the last 4 byte to the main memory
+ 0x0C     0x80000004
```

Transfer sequence to PM_FIFO is as follows.

1) GOKU-S transfer (write) the first 1020 (=1024-4) byte to PM_FIFO.
2) GOKU-S transfer (write) the last 4 byte to the main memory.
3) GOKU-S read the last 4 byte which is transferred to the main memory.
4) The software copy (write) the last 4 byte which is transferred to the main memory to PM_FIFO.

(b) USB Host Controller

Adding the dummy TD (Transfer Descriptor) same as case of Bus Master IDE Controller.

14.9 Electrostatic Discharge

[Electrostatic Discharge Testing Results]

The following table shows the results of Electrostatic Discharge testing that were performed on this device.

When handling individual devices (which are not yet mounted on a printed circuit board), be sure that the environment is protected against electrostatic electricity

Please refer to “General Safety Precautions and Usage Considerations” section of the data book for more information.

Standard	Pin(s)	Rated Voltage
Machine Model (ESD-Association S5.2 (JEDEC JESD22A-115A))	ALL pins	200 V or more
Human Body Model (ESD-Association S5.1 (JEDEC JESD22A-114B))	RESET*, DD[15:0] , DIOR*, DIOW*, DMARQ*, IORDY, DA[2:0], DMACK*, CS0*, CS1*, INTRQ, PDIAG*	1000V
	Other pins	2000 V or more

14.10 Limitation on the Set Command IO Base Address Register

- Relevant block: Bus Master IDE Controller (Function 0)

[Limitation]

When Command IO Base Address Register (0x10) is written, D3 bit must be set to 0

[Symptom]

In the Specification of PCI IDE Controller Specification Revision 1.0, Base registers used to map Command Block registers must ask for 8 bytes of IO space. So, originally Command IO Base Address Register (0x10) used to map Command Block registers should be allowed an address setup by 8-byte alignment.

However, if D3 bit of Command IO Base Address Register (0x10) is written to 1 (like 0xXXXXXXX8), the value of upper 8 bits read from (or written to) the Data Register of 16-bit width will surely be set to 0x00.

Example (Command IO Base Address=0xa000108)

original data:	0xABCD
read data in this problem:	0x00CD

[Workarounds]

When Command IO Base Address Register (0x10) is written, you should perform an address setup by 16-byte alignment so that D3 bit is set to 0.

14.11 Limitation when the USB Device Controller Communicates to the USB Host through a Bulk or Interrupt OUT Transfer

- Relevant block: USB Device Controller (Function 2)

[Outline]

When the USB Device Controller (UDC) communicates to the USB host through a bulk or interrupt OUT transfer, the UDC may show a different behavior from the USB 1.1 specification.

[Symptoms]

When the USB host fails to receive on ACK from the USB Device Controller (UDC) in response to an OUT transfer, the USB host resends the same data. If the FIFO in the UDC is ready to accept data, the UDC receives the same packet with the same data toggle as for the previous packet. Thus, the UDC detects a toggle error and returns an ACK to the USB host. The UDC discards this packet because it has already received it normally.

If the FIFO is full at this time, the UDC returns a NAK to indicate to the USB host that it is not ready to receive a new packet.

<USB 1.1 Specification>

Section 8.4.5.3, "Function Response to an OUT Transaction," of the USB 1.1 specification shows the possible responses a function may make in response to an OUT transaction in order of precedence. It explicitly states that an ACK response due to a data toggle error (SequenceBitsMatch = No) is given precedence over a NAK response.

Table 14.11.1 Function Responses to OUT Transactions in Order of Precedence

Data Packet Corrupted	Receiver Halt Feature	Sequence Bits Match	Function Can Accept Data	Handshake Returned by Function
Yes	N/A	N/A	N/A	None
No	Set	N/A	N/A	STALL
No	Not set	No	N/A	ACK
No	Not set	Yes	Yes	ACK
No	Not set	Yes	No	NAK

However, the GOKU-S UDC gives priority to the check of the FIFO state over toggle error detection and returns a NAK to the USB host if its FIFO can not accept a packet for some reason (e.g., FIFO full). This is discussed in Section 8.1.4.2.3, "Bulk reception mode," of this databook along with a transaction flowchart. Thus, when the FIFO is not ready, the UDC action deviates from the USB specification.

[Cause]

The circuit block in the UDC that controls handshake responses gives priority to a check of the FIFO state over toggle error detection and generates a handshake to the USB host.

[Influence]

A retry of a missed packet due to a data toggle error should normally complete with a single transaction as per the USB 1.1 specification. However, the GOKU-S UDC might generate multiple transactions, depending on the FIFO state.

More specifically, if the UDC receives a data packet with a toggle error when there is no empty space available in the FIFO, it returns a NAK. When the FIFO becomes ready to accept a packet, the UDC then returns an ACK to the USB host. This completes a retry.

If the FIFO_DISABLE command is issued to the endpoint that is transferring data, the UDC returns a NAK to the USB host. When the UDC FIFO becomes ready after the FIFO_DISABLE command is cancelled (via a FIFO_ENABLE command), the UDC returns an ACK, completing a retry operation.

14.12 USB Host Controller's Limitation on Isochronous OUT Transfers

- Relevant block: USB Host Controller (Function 1)

[Symptom]

When a data underrun condition is detected, the USBHC sets the underrun flag. After that, the transmit FIFO once becomes empty. If the immediately following transmit data is 0x7e******, the internal signal that indicates the byte boundary in the transmit data and the FIFO empty flag get stuck, causing the internal states to also get stuck. If this happens, the USBHC ceases to issue SOF, and consequently, the device moves to the Suspend state or becomes disconnected.

[Limitation]

Typically, isochronous OUT transfers are used only by USB speakers. The above problem does not arise if no data underrun occurs. When isochronous OUT transfers are used, care must be taken so that no data underrun will occur.

14.13 USB Host Controller's Limitation on Receiving Short Packets

- Relevant block: USB Host Controller (Function 1)

[Symptom]

There are peculiar USB systems where once the USB enters the BUSY state, the host might not be able to gain bus mastership for a long period of time. The length of the write interval, defined to be 1 ms in the USB specification, could become 2 to 7 ms. If the USB 1.1 host is used in such a system, the host might fail to perform a status writeback properly, causing a data overrun.

[Limitation]

When the OHCI core is active and receives a short packet in response to an IN transfer issued by the host, the host will skip some of the status writeback steps under the following conditions:

Condition 1: When receiving a short packet

Condition 2: When the Current Buffer Pointer (CBP) in a General Transfer Descriptor (GTD) does not reach the Buffer End (BE) during a short-packet reception. (The CBP points to the current receive data address that is used to write the receive data to memory. The BE represents the end of the buffer allocated to store the receive data.) No problem occurs when the CBP reaches the BE.

Condition 3: When the VM_ACK signal to the host core is not asserted for a long period due to the USB system being in the BUSY state, and the period during which VM_ACK is in the deasserted state goes beyond the next SOF. (In other words, when the host core is forced to wait for such a long period that it can not complete a status writeback within the same frame.)

If a short-packet transfer completes normally as scheduled in the descriptor, a status writeback consists of the following three steps. However, when the above conditions are true, steps 2 and 3 will be skipped.

1. The CBP for the receive data is moved forward.
2. The link to the Next TD is updated.
3. The link to the Done Queue is updated.

Skipping steps 2 and 3 causes a data overrun. Consequently, the data overrun flag in the transfer descriptor is set, and the Halt bit in the endpoint descriptor is set.

[Limitation]

Please use the USBHC in a USB system in which the frame number is updated every 1 ms. (Usually, this is necessary according to the USB specification to ensure consistency in the frame time between hardware and software as required in isochronous transfers.)

If a problem occurs in a USB system in which frame number updates are not guaranteed to occur every 1 ms, a software recovery can be achieved as follows using a class driver:

1. Discard all data received through the descriptor halted due to a data overrun, clear the Halted bit, reschedule the endpoint descriptor and receive data from the start again.
2. Treat the error as unrecoverable and apply a USB Reset.

14.14 USB Host Controller's Limitation on Resume Signaling

- Relevant block: USB Host Controller (Function 1)

[Symptom]

When a remote wakeup event occurs at a downstream port, the OHCI host begins to send the resume signal to the device. If the driver sets the HcControl.HCFS field in the OHCI operational register to USBOPERATIONAL during the resume signaling, the host stops sending the resume signal. Consequently, the device fails to recognize the resume signaling and remains in the Suspend state.

[Solution]

The driver must wait for 20 ms or longer before it changes the HcControl.HCFS field from USBRESUME to USBOPERATIONAL.

According to the USB specification, the USB device shall continue to receive the resume signaling from the host for at least 20 ms. Usually, the device driver allows this period before setting HcControl.HCFS to USBOPERATIONAL. (We have confirmed that VxWorks 5.5.1 (PCD 1.1) and MontaVista Linux 3.1 satisfy this requirement.)

TC86C001FG Revision History

Page	Rev. 2.0	Rev. 2.1
4-1 4-2 4-3 4-4	—	Added the PU (pull-up resistors) and OD (open drain) designators to the Pin Type Column
12-2	—	Added the TRST* signal to (4)

X-ON Electronics

Largest Supplier of Electrical and Electronic Components

Click to view similar products for I/O Controller Interface IC ***category:***

Click to view products by Toshiba ***manufacturer:***

Other Similar products are found below :

DSL4510 S R15X DSL5110 SR1TY EC-GAV SEC1210I-CN-02 LPC47M107S-MS LPC47M102S-MS 70M-OAC15A IS31IO7326-QFLS4-EB PM8001C-F3EI SLO24IRA LPC47B277-MS JHL8040R S LMN6 DS2488X+T BU92747GUW-E2 IDC5Q FDC37B787-NS PCI1520IPDVEP MCP2140A-I/P CQM1-LK501 IDC-24F OAC15 ODC15 OAC24 OAC24A MCP2140A-ISO OAC5A IA82510PLC28IR2 MAX5942BCSE+ 70G-IAC15 70M-ODC15B DSL2310 S LJ3W JHL6240 S LLNG JHL7340 S LMHX JHL7540 S LMHR JHL7440 S LMHZ JHL8540 S RH4Q JHL8340 S RH4N NH82801IB S LA9M MAX5940CESA+ DS2488X+U DS8005-RJX+ MCP2140A-ISS MCP2150-I/SS MCP2155-I/SS MCP2140AT-I/SS MCP2140-I/SS DS2484R+T MAX5940DESA+ PCA9541APW/03,118 LPC47N217N-ABZJ