Atmel ATPL210A

PRIME compliant Power Line Communications SoC

DATASHEET

Features

- Core
  - ADD8051C3A enhanced 8051 core
  - Speedups up to x5 vs. standard 8051 microcontroller
- Modem
  - Power Line Carrier Modem for 50 and 60 Hz mains
  - 97-carrier OFDM PRIME compliant
  - Baud rate Selectable: 21400 to 128600 bps
  - Differential BPSK, QPSK, 8-PSK modulations
- Memories
  - 32Kbytes on-chip SRAM
  - Up to 256Kbytes external SRAM
- In-circuit serial flash programming
- Auto boot-loading program from serial flash
- Automatic Gain Control and signal amplitude tracking
- Embedded on-chip DMAs
- Automatic code encryption during boot loading
- Media Access Control
  - Viterbi decoding and CRC PRIME compliant
  - 128-bit AES encryption
  - Channel sensing and collision pre-detection
- Peripherals
  - Two 2-wire UARTs
  - Two SPI. SPI to serial flash and External RTC. Buffered SPI to external metering IC
  - Programmable Watchdog
  - Up to 14 I/O lines
- Package
  - 120-lead LQFP, 14 x 14 mm, pitch 0.4 mm
  - Pb-free and RoHS compliant
- Typical Applications
  - Automated Meter Reading (AMR) & Advanced Meter Management (AMM)
  - Street lighting
  - Home Automation

Description

The ATPL210A is a Power Line Communications System on Chip, which implements a full PRIME compliant PLC modem. It includes an enhanced 8051 microcontroller (IP core ADD8051C3A), a Medium Access Controller (MAC) (IP core ADD1221) and a Modem circuit (IP core ADD1321) for power line medium using OFDM modulation compatible with PRIME specifications.

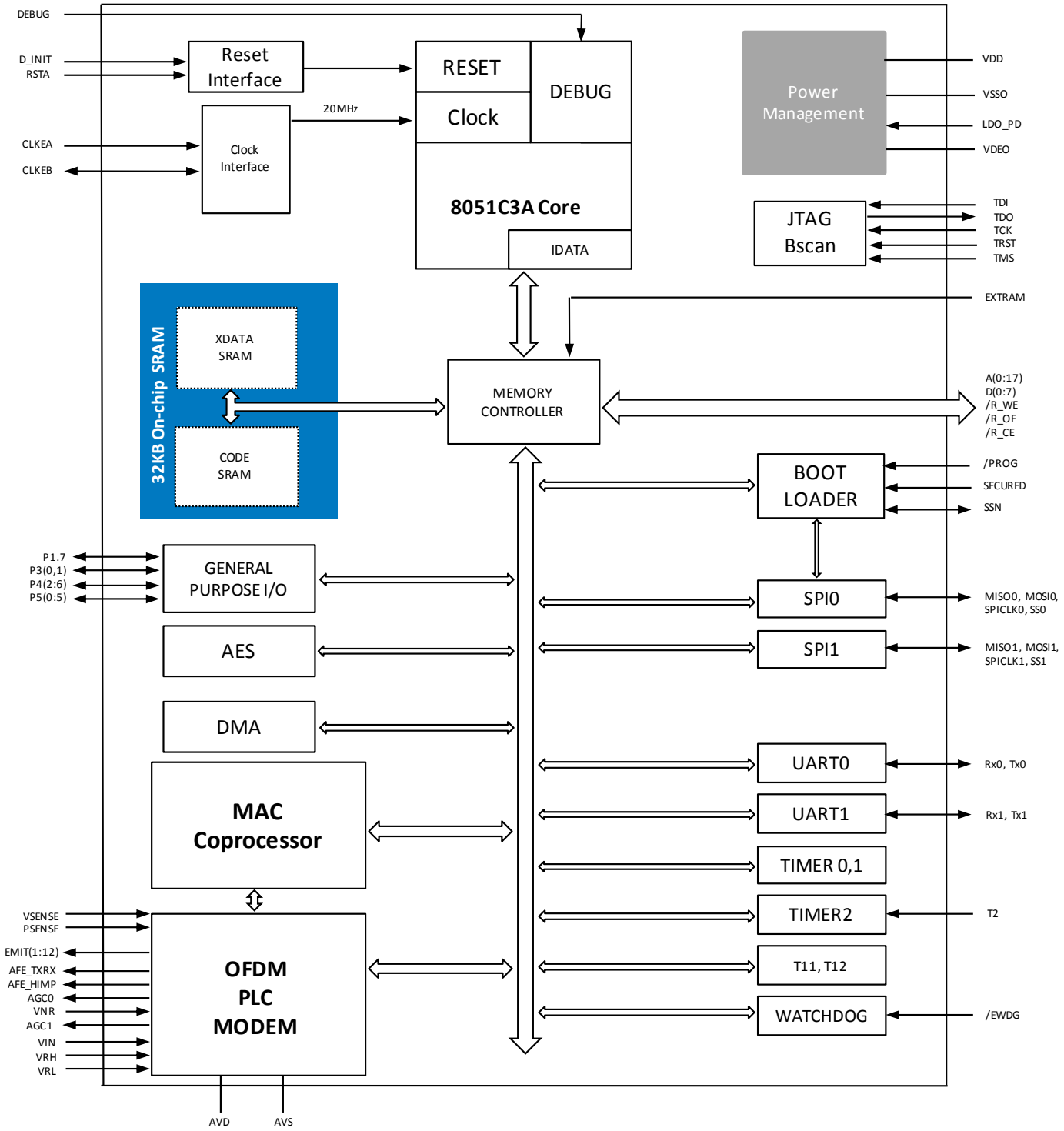
ATPL210A is oriented to high performance & robust AMR systems. The ATPL210A is designed to be used by meter manufacturers to provide a low cost and compact solutions for AMR (Automated Meter Reading) & AMM (Advanced Meter Management) systems using narrow band power line communications.

This device has been developed to reduce CPU computational load in PLC systems running PRIME protocols. ATPL210A includes all necessary resources to be used as main controller in metering applications, and allows an external device to communicate according to PLC PRIME specifications.

Table of Contents

1. Block Diagram

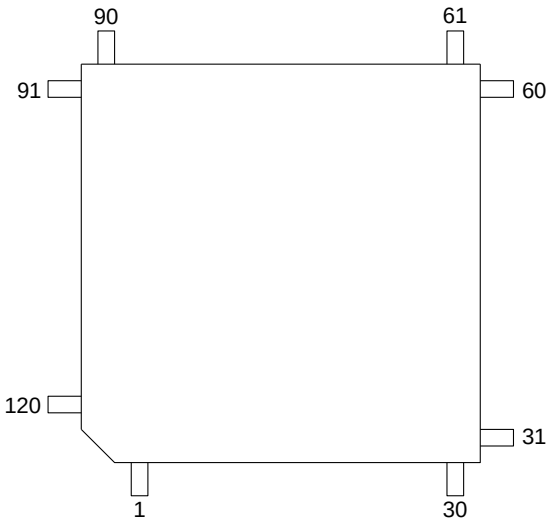
Figure 1-1. ATPL210A 120-pin Block Diagram



2. Package and Pinout

2.1 120-Lead LQFP Package Outline

Figure 2-1. Orientation of the 120-Lead Package



2.2 120-Lead LQFP Pinout

Table 2-1. ATPL210A 120-Lead LQFP pinout

PinNo	Pin Name	I/O	I(mA)	Res	HY
1	A17	O	±2	-	-
2	A9	O	±2	-	-
3	GND	P	-	-	-
4	VCC	P	-	-	-
5	/R_WE	O	±2	PU	-
6	D4	I/O	±2	-	-
7	D3	I/O	±2	-	-
8	D5	I/O	±2	-	-
9	D2	I/O	±2	-	-
10	D6	I/O	±2	-	-
11	D1	I/O	±2	-	-
12	D7	I/O	±2	-	-
13	D0	I/O	±2	-	-
14	VCC	P	-	-	-
15	GND	P	-	-	-
16	VDD	P	-	-	-
17	/R_OE	O	±2	-	-
18	/R_CE	O	±2	-	-
19	A8	O	±2	-	-
20	A0	O	±2	-	-
21	A7	O	±2	-	-
22	A1	O	±2	-	-
23	A6	O	±2	-	-
24	A2	O	±2	-	-
25	A5	O	±2	-	-
26	GND	P	-	-	-
27	VCC	P	-	-	-
28	A3	O	±2	-	-
29	A4	O	±2	-	-
30	P4.5/MISO1	I/O	±2	PU	-
31	P4.4/MOSI1	I/O	±2	PU	-
32	P4.3/SPICLK1	I/O	±2	PU	-
33	P4.2/SS1	I/O	±2	PU	-
34	P3.0/RxD0	I/O	±2	PU	-
35	P3.1/TxD0	I/O	±2	PU	-
36	VCC	P	-	-	-
37	GND	P	-	-	-
38	EMIT.1	O	±X	-	-
39	EMIT.2	O	±X	-	-
40	EMIT.3	O	±X	-	-
41	EMIT.4	O	±X	-	-
42	VCC	P	-	-	-
43	GND	P	-	-	-
44	EMIT.5	O	±X	-	-
45	EMIT.6	O	±X	-	-
46	EMIT.7	O	±X	-	-
47	EMIT.8	O	±X	-	-
48	VCC	P	-	-	-
49	GND	P	-	-	-
50	EMIT.9	O	±X	-	-
51	EMIT.10	O	±X	-	-
52	EMIT.11	O	±X	-	-
53	EMIT.12	O	±X	-	-
54	VCC	P	-	-	-
55	GND	P	-	-	-
56	AFE_HIMP	O	±X	-	-
57	AFE_TXRX	O	±X	-	-
58	VSENSE	I	-	-	Y
59	PSENSE	I	-	-	Y
60	VNR	I	-	-	Y
61	TDI	I	-	PU	-
62	TMS	I	-	PU	-
63	TDO	O	±2	-	-
64	GND[1]	P	-	-	-
65	GND	P	-	-	-
66	VCC	P	-	-	-
67	TRST	I	-	PU	-
68	TCK	I	-	-	-
69	RSTA	I	-	PD	Y
70	D_INIT	I	-	PD	Y
71	GND	P	-	-	-
72	VCC	P	-	-	-
73	GND	P	-	-	-
74	VDD	P	-	-	-
75	LDO_PD	I	-	-	-
76	VSS0	P	-	-	-

PinNo	Pin Name	I/O	I(mA)	Res	HY
77	VDE0	P	-	-	-
78	VDE0	P	-	-	-
79	GND	P	-	-	-
80	GND	P	-	-	-
81	VCC	P	-	-	-
82	CLKEA	I	-	-	-
83	GND	P	-	-	-
84	CLKEB	I/O	-	-	-
85	VCC	P	-	-	-
86	/EWDG	I	-	PD	Y
87	DEBUG	I	-	PD	Y
88	EXTRAM	I	-	-	Y
89	/PROG	I	-	PU	Y
90	SECURED	I	-	PD	Y
91	P5.4/RxD1	I/O	±2	PU	-
92	P5.5/TxD1	I/O	±2	PU	-
93	NC	-	-	-	-
94	P4.6/T2/AGC1	I/O	±X	PU	-
95	AGC0	O	±X	-	-
96	GND	P	-	-	-
97	VCC	P	-	-	-
98	AVS2	P	-	-	-

PinNo	Pin Name	I/O	I(mA)	Res	HY
99	AVD2	P	-	-	-
100	AVS1	P	-	-	-
101	AVD1	P	-	-	-
102	VRH	I	-	-	-
103	VIN	I	-	-	-
104	VRL	I	-	-	-
105	GND	P	-	-	-
106	VCC	P	-	-	-
107	P5.0/SS0	I/O	±2	PU	-
108	P5.3/MISO0	I/O	±2	PU	-
109	P5.2/MOSI0	I/O	±2	PU	-
110	P5.1/SPICLK0	I/O	±2	PU	-
111	P1.7/SSN	I/O	±2	PU	-
112	A13	O	±2	-	-
113	A14	O	±2	-	-
114	GND	P	-	-	-
115	VCC	P	-	-	-
116	A12	O	±2	-	-
117	A15	O	±2	-	-
118	A11	O	±2	-	-
119	A16	O	±2	-	-
120	A10	O	±2	-	-

Notes: 1. Mandatory to be tied down

I/O=pin direction: **I**=input, **O**=Output, **P**=Power

I(mA)=nominal current: **+**=source, **-**=sink, **X**=fixed by external resistor

RES=pin pullup/pulldown resistor: **PU**=pullup, **PD**=pulldown,

HY=Input Hysteresis

3. Pin Description

Table 3-1. Pin Description List

Pin Number	Pin Name	Type	Comments
1, 2, 19, 20, 21, 22, 23, 24, 25, 28, 29, 112,113, 116, 117, 118, 119, 120	A(0:17)	Output	External Memory data/program Address bus. • This bus is enabled when the system is configured to work with an external memory device (EXTRAM pin = ’1’). If the system is configured to work with the on-chip SRAM configuration (EXTRAM pin = ’0’), then these pins output is ’0’.
3, 15, 26, 37, 43, 49, 55, 64, 65, 71, 73, 79,80, 83, 96, 105, 114	GND	Power	Digital Ground
4, 14, 27, 36, 42, 48, 54, 66, 72, 81, 85, 97,106, 115	VCC	Power	3.3v digital supply. Digital power supply must be decoupled by external capacitors
5	/R_WE	Output	External Memory Write Enable signal. • This output is enabled when the system is configured to work with an external memory device (EXTRAM=’1’). If the system is configured to work with the on-chip SRAM configuration (EXTRAM=’0’), then this output must be left unconnected. • /R_WE is active low • Internal configuration: 33kΩ typ. pull-up resistor
6, 7, 8, 9, 10, 11, 12, 13	D(0:7)	I/O	External Memory Data bus. • This bus is enabled when the system is configured to work with an external memory device (EXTRAM=’1’). If the system is configured to work with the on-chip SRAM (EXTRAM=’0’), then these pins are in Hi-Z.
16, 74.	VDD	Power	LDO Power Output. A capacitor in the range 0.1µF-10µF must be connected to each pin
17	/R_OE	Output	External Memory Output Enable signal. • This output is enabled when the system is configured to work with an external memory device (EXTRAM=’1’). If the system is configured to work with the on-chip SRAM configuration (EXTRAM=’0’), then this pin output is ‘1’. • /R_OE is active low
18	/R_CE	Output	External Memory Chip Enable signal. • This output is enabled when the system is configured to work with an external memory device (EXTRAM=’1’). If the system is configured to work with the on-chip SRAM configuration (EXTRAM=’0’), then this pin output is ‘1’. • /R_CE is active low

Pin Number	Pin Name	Type	Comments
30	P4.5/MISO1	I/O	Microcontroller port 4.5 / SPI1 Master In Slave Out. • When configured as P4.5, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as MISO1, this pin is the SPI1 Master In Slave Out • Internal configuration: 33kΩ typ. pull-up resistor
31	P4.4/MOSI1	I/O	Microcontroller port 4.4 / SPI1 Master Out Slave In • When configured as P4.2, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as MOSI1, this pin is the SPI1 Master Out Slave In • Internal configuration: 33kΩ typ. pull-up resistor
32	P4.3/SPICLK1	I/O	Microcontroller port 4.3 / SPI1 Clock • When configured as P4.3, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as SPICLK1, this pin is the SPI1 clock signal • Internal configuration: 33kΩ typ. pull-up resistor
33	P4.2/SS1	I/O	Microcontroller port 4.2 / SPI1 Slave Select • When configured as P4.2, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as SS1, this pin is the SPI1 Slave Select. Active low • Internal configuration: 33kΩ typ. pull-up resistor
34	P3.0/RxD0	I/O	Microcontroller port 3.0 / Standard Serial Port 0 Rx • When configured as P3.0, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as RxD0, this pin is the digital input of the asynchronous standard serial port 0 Internal configuration: 33kΩ typ. pull-up resistor
35	P3.1/TxD0	I/O	Microcontroller port 3.1 / Standard Serial Port 0 Tx • When configured as P3.1, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as TxD0, this pin is the digital output of the asynchronous standard serial port 0 Internal configuration: 33kΩ typ. pull-up resistor
38, 39, 40, 41, 44, 45, 46, 47, 50, 51, 52, 53	EMIT(1:12)	Output	PLC Transmission ports(1)

Pin Number	Pin Name	Type	Comments
56	AFE_HIMP	Output	Analog Front-End High-Impedance • This digital output is used by the chip to select between low-impedance and high-impedance transmission branch (when working with a “two half-H-bridge branches” analog front end configuration). This way, the system adapts its transmission external circuitry to the net impedance, improving transmission behavior. The polarity of this pin can be inverted by hardware. Please refer to the Reference Design for further information.
57	AFE_TxRx	Output	Analog Front-End Transmission/Reception • This digital output is used to select between external Transmission and Reception branches. The suitable value depends on the external circuitry configuration. The polarity of this pin can be inverted by hardware. Please refer to the Reference Design for further information.
58	VSENSE	Input	Voltage Level Sensing • This input tracks the voltage level in the power supply to avoid power supply malfunction.
59	PSENSE	Input	Power Level Sensing • This input tracks the power level in the power supply to avoid power supply malfunction.
60	VNR	Input	Zero Crossing Detection Signal • This input detects the zero-crossing of the mains voltage, needed to determine proper switching times. Depending on whether an isolated or a non-isolated power supply is being used, isolation of this pin should be taken into account in the circuitry design. Please refer to the Reference Design for further information.
61	TDI(2)	Input	Test Data In • Internal configuration: 33kΩ typ. pull-up resistor
62	TMS(2)	Input	Test Mode Select • Internal configuration: 33kΩ typ. pull-up resistor
63	TDO(2)	Output	Test Data out
67	TRST(2)	Input	Test Reset • Internal configuration: 33kΩ typ. pull-up resistor
68	TCK(2)	Input	Test Clock
69	RSTA	Input	Asynchronous reset • RSTA is a digital input pin used to perform a hardware reset of the ASIC • RSTA is active high • Internal configuration: 33kΩ typ. pull-down resistor

Pin Number	Pin Name	Type	Comments
70	D_INIT	Input	Initialization Signal • During power-on, D_INIT should be released before asynchronous reset signal RSTA, in order to ensure proper system start up. Not minimum time is required between both releases, $\Delta t>0$ • D_INIT is active high • Internal configuration: 33kΩ typ. pull-down resistor

75	LDO_PD	Input	LDO Power-down
			• This digital input is used to put the internal linear regulator into power down mode
			  • '0': Power down mode disabled
			  • '1': Power down mode enabled

76	VSS0	Power	LDO ground
77, 78	VDE0	Power	LDO 3.3v power supply
82	CLKEA[3]	Input	External clock reference • CLKEA must be connected to one terminal of a crystal (when a crystal is being used) or tied to ground if a compatible oscillator is being used
84	CLKEB[3]	I/O	External clock reference • CLKEB must be connected to one terminal of a crystal (when a crystal is being used) or to one terminal of a compatible oscillator (when a compatible oscillator is being used)

86	/EWDG	Input	Watchdog enable
			• /EWDG digital input enables watchdog timer. This pin is internally connected to the /EW signal of the ADD8051C3A microcontroller
			  • '0': Watchdog timer enabled
			  • '1': Watchdog timer disabled
			• Internal configuration: 33kΩ typ. pull-down resistor

87	DEBUG	Input	Debug mode enable
			• DEBUG digital input is internally connected to DBG signal of the ADD8051C3A microcontroller, and it is intended to implement software debugging tools
			  • '0': Debug mode disabled
			  • '1': Debug mode enabled
			• Internal configuration: 33kΩ typ. pull-down resistor

Pin Number	Pin Name	Type	Comments

88	EXTRAM	Input	SRAM device selection
			• This digital input allows selecting between working with an external SRAM device or with the internal SRAM embedded on chip.
			  ◦ '0': 32Kbytes on-chip SRAM selected.
			  ◦ '1': External memory device selected. (The on-chip SRAM is not accessible when working in this mode).

89	/PROG	Input	SPI Flash programming pin
			• /PROG digital input is read during power up and sets the system into "execution" mode or "serial flash programming" mode
			  ◦ '0': Serial flash programming mode
			  ◦ '1': Execution mode (normal mode)
			• Internal configuration: 33kΩ typ. pull-up resistor

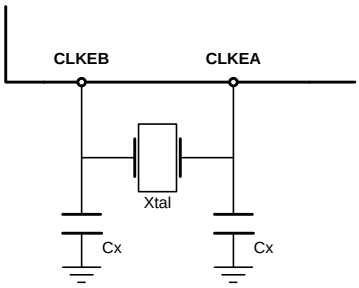
90	SECURED[4]	Input	Encryption enable
			• SECURED digital input enables encrypted firmware storage and execution when the board configuration supports it
			  ◦ '0': Encrypted storage/execution disabled
			  ◦ '1': Encrypted storage/execution enabled
			• Internal configuration: 33kΩ typ. pull-down resistor

91	P5.4/RxD1	I/O	Microcontroller port 5.4 / Standard Serial Port 1 Rx • When configured as P5.4, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as RxD1, this pin is the digital input of the asynchronous standard serial port 1 • Internal configuration: 33kΩ typ. pull-up resistor
92	P5.5/TxD1	I/O	Microcontroller port 5.5 / Standard Serial Port 1 Tx • When configured as P5.5, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as TxD1, this pin is the digital output of the asynchronous standard serial port 1 • Internal configuration: 33kΩ typ. pull-up resistor
93	NC		No connect

Pin Number	Pin Name	Type	Comments
94	P4.6/T2/AGC1	I/O	Microcontroller port 4.6/T2/AGC1 • When configured as P4.6, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as T2, this pin works as the external T2 pin described in Timer2 section • When configured as AGC1, this pin is managed by AGC hardware logic to drive external circuitry if input signal attenuation is needed • Internal configuration: 33kΩ typ. pull-up resistor
95	AGC0	Output	Automatic Gain Control 0 • This digital output is managed by AGC hardware logic to drive external circuitry if input signal attenuation is needed
98, 100	AVS1, AVS2	Power	Analog ground
99, 101	AVD1, AVD2	Power	3.3v analog power
102	VRH[5]	Input	Analog input high voltage reference
103	VIN[5]	Input	Direct-analog input voltage
104	VRL[5]	Input	Analog input low voltage reference
107	P5.0/SS0	I/O	Microcontroller port 5.0 / SPI0 Slave Select • When configured as P5.0, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as SS0, this pin is the SPI0 Slave Select. Active low • Internal configuration: 33kΩ typ. pull-up resistor
108	P5.3/MISO0	I/O	Microcontroller port 5.3 / SPI0 Master In Slave Out • When configured as P5.3, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as MISO0, this pin is the SPI0 Master In Slave Out • Internal configuration: 33kΩ typ. pull-up resistor
109	P5.2/MOSI0	I/O	Microcontroller port 5.2 / SPI0 Master Out Slave In • When configured as P5.2, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as MOSI0, this pin is the SPI0 Master Out Slave In • Internal configuration: 33kΩ typ. pull-up resistor

Pin Number	Pin Name	Type	Comments
110	P5.1/SPICLK0	I/O	Microcontroller port 5.1 / SPI0 Clock • When configured as 5.1, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as SPICLK0, this pin is the SPI0 clock signal • Internal configuration: 33kΩ typ. pull-up resistor
111	P1.7/SSN	I/O	Microcontroller port 1.7 / Silicon Serial Number • When configured as P1.7, this pin is a pseudo-bidirectional microcontroller I/O port • This pin is the digital input used to read a Serial number if a valid SSN device is being used. This Serial Number is used for encryption purposes. Precaution should be taken if used as generic control port since it searches for a Silicon Serial Number device at start-up and could put out undesirable transient values • Internal configuration: 33kΩ typ. pull-up resistor

Notes:
1. Different configurations allowed depending on external topology and net behavior
2. This pin is part of the JTAG Boundary Scan interface and is only used for boundary scan purposes
3. The crystal should be located as close as possible to CLKEA and CLKEB pins. Recommended value for Cx is 18pF. This value may depend on the specific crystal characteristics



4. See supported devices section 10.4.3
5. See Reference Design for suitable values

4. PRIME overview

The PRIME (PoweRline Intelligent Metering Evolution) initiative was originally conceived as an answer to the need for a future-proof, cost-effective Automatic Meter Management (AMM) solution. During its evolution in the last two years, it has evolved into a solution for an entire Smart Grid environment which will contribute definitively to energy efficiency improvement and ultimately to addressing the pressing issue of climate change.

The overall performance of the system fully complies with requirements for a new, stable metering infrastructure that is set to become a fundamental part of the Smart Grids. The revolutionary openness of the PRIME solution and its focus on interoperability as a way to decrease costs in a competitive framework, has been arising an unprecedented level of interest within the industry and markets. For this very reason a decision was taken by several of the most committed parties to set up the PRIME Alliance.

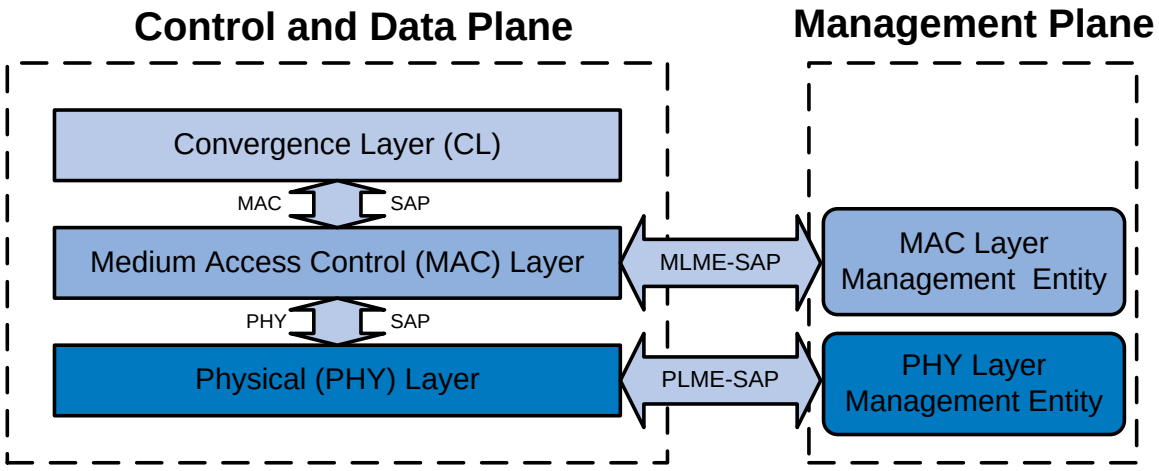
PRIME Alliance remains unique in that it shuns proprietary technologies and is the result of an open, cost-oriented effort among many different partners seeking to develop a future market for common benefit. Royalties and patents are definitely not part of the core PRIME specification. Ownership of the PRIME specification is uniquely public.

4.1 Key Features

PRIME defines lower layers of a PLC narrowband data transmission system over the electric grid. All the system has been created to be low cost and high performance.

Figure 4-1 below depicts the proposed communication layers and the scope of the specification. The proposed reference model is based on IEEE Std. 802.16 protocol layering.

Figure 4-1. PRIME layers



The service-specific **Convergence Layer** (CL) classifies traffic associating it with its proper MAC connection. This layer performs the mapping of any kind of traffic to be properly included in MAC SDUs (Service Data units). It may also include payload header suppression functions. Multiple Convergence sub-layers s are defined in order to accommodate different kinds of traffic into MAC SDUs.

The **MAC layer** provides core MAC functionalities of system access, bandwidth allocation, connection management and topology resolution. It has been defined for a connection oriented Master-Slave environment, and optimized for low voltage power line environments.

The **PHY layer** transmits and receives MAC PDUs (Protocol Data Units) between Neighbor Nodes. It is based on OFDM multiplexing in CENELEC A band and reaches up to 130 kbps raw data rate.

PRIME specifications take advantages of state of the art technologies and adapt them to the needed requirements, simplifying processes, overheads and others, to ensure performance, interoperability between devices and different implementations of elements in the system.

4.2 PRIME physical layer overview

4.2.1 Main features

PRIME PHY layer is designed to transmit and receive over power lines which were originally devised for distribution of power at 50-60Hz AC. The use of this medium for communications at higher frequencies presents some challenging technical problems:

- Distribution networks are usually made of a random variety of conductor types, and terminating into loads of different impedances. Such a network has an amplitude and phase response that varies widely with frequency. Furthermore, the channel characteristics shall also vary with time as the loads on the network change.
- Interference also affects power lines. Electric appliances with different kind of motors, switching power supplies and halogen lamps produce impulse noise that reduces the reliability of communication signals. Due to attenuation, the noise is also location dependent.

PRIME PHY layer uses a combination of approaches that ultimately allow for robust high speed, low cost communications over power lines. A simple yet powerful scheme which is based on adaptively modulated Orthogonal Frequency Division Multiplexing (OFDM), along with forward error correction and data interleaving.

A block diagram representation of a PHY transmitter is shown below in Figure 4-2:

Figure 4-2. PHY transmitter

On the transmitter side, the PHY layer receives its inputs from the Media Access Control layer. If decided by higher layers, the PHY frame after the CRC block is convolutionally encoded and interleaved; however, it will always be scrambled). The output is differentially modulated using a DBPSK, DQPSK or D8PSK scheme. The next step is OFDM, which comprises the IFFT (Inverse Fast Fourier Transform) block and the cyclic prefix generator.

4.2.2 Orthogonal Frequency Division Multiplexing (OFDM)

One of the main novelties of PRIME is that it uses an OFDM approach instead of traditional single carrier solutions that have been used in the past for narrowband power line communications.

OFDM is well known in the literature and in industry. It is currently used in xDSL technologies, terrestrial wireless distribution of television signals (DVB-T, DVB-H and more), and has also been adapted for IEEE's high rate wireless LAN Standards (802.11a and 802.11g). In less than twenty years, in fact, OFDM has developed into a popular scheme for virtually all new telecoms standards: WiMAX, DAB, DRM, 3G cellular telephony, UWB, MoCA, Broadband over Power line...

OFDM is a digital multi-carrier modulation scheme, which uses a large number of closely-spaced orthogonal subcarriers to carry data. To obtain high spectral efficiency, these subcarriers typically overlap in frequency. However the mathematical property of orthogonality allows recovering each of the subcarriers separately at the receiver, so in practice subcarriers do not interfere with each other as would be the case with traditional FDM.

Each subcarrier is modulated with a conventional modulation scheme at a low symbol rate, maintaining data rates similar to conventional single-carrier modulation schemes in the same bandwidth. In practice, OFDM signals are efficiently generated and detected using the well-known Fast Fourier Transform (FFT) algorithm.

The primary advantage of OFDM over single-carrier schemes is its ability to cope with severe channel conditions -for example, attenuation of high frequencies in long power lines, narrowband interference and frequency-selective fading due to multipath- without complex additional mechanisms (e.g. equalization filters). Channel equalization is simplified because OFDM may be viewed as using many slowly-modulated narrowband signals rather than one rapidly-modulated wideband signal.

Additionally, low symbol rate makes the use of a guard interval (or cyclic prefix) between symbols affordable, rendering it possible to handle time-spreading and eliminate intersymbol interference (ISI). For low frequencies like the ones PRIME uses, multipath is not a critical issue so cyclic prefixes will not waste a significant part of OFDM symbols.

4.3 PRIME MAC layer overview

PRIME system is composed of sub networks, each of them defined in the context of a transformer station. A sub network is a tree with two types of nodes, the Base Node and the Services Nodes.

4.3.1 Base Node

The Base Node is at the root of the tree and acts as master node that provides connectivity to the sub network. It manages the sub network resources and connections.

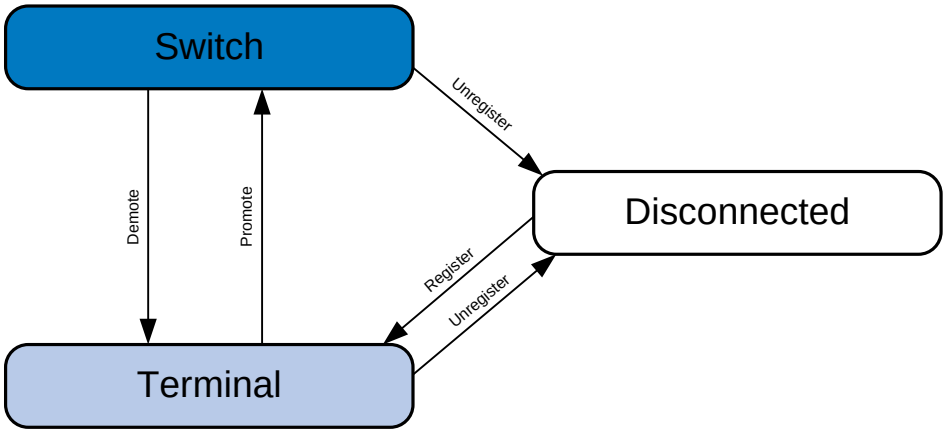
There is only one Base Node in a sub network. This Base Node is initially the sub network itself, and other nodes should follow a process of registering in order to enroll them to this sub network.

4.3.2 Service Node

Any other node of the sub network is a Service Node. Service Nodes are either leaves of the tree or branch points of the tree. These nodes start in a disconnected state and follow certain procedures to establish network connectivity. Each of these nodes is one point of the mesh of the sub network. These nodes have two responsibilities: connecting themselves to the sub network and switching the data of their neighbors in order to propagate connectivity.

Service Nodes change their behavior dynamically from “Terminal” functions to “Switch” functions and vice-versa. Changing of functional states occurs based on certain predefined events in the network.

Figure 4-3. Functional states of a Service Node



As shown in Figure 4-3, the three functional states of a Service Node are:

- **Disconnected**: Service Nodes start in a disconnected state. In this state a node is not capable of communicating or switching the traffic of another node. The primary function of a Service Node in this state is to search for an operational network in its proximity and to try to register itself to it.
- **Terminal**: In this state a Service Node is capable of communicating its traffic by establishing connections, but is not capable of switching the traffic of any other node.
- **Switch**: In this state a Service Node is capable of performing all Terminal functions. Additionally, it is capable of forwarding data to and from other devices in the sub network. It is a branch point in the tree.

The MAC layer provides all necessary features to manage PRIME networks and sub networks: addressing, synchronization (beacon management), dynamic management of the network structure (promotion and demotion of terminals), device registration management, connection setup and management, channel access arbitration, distribution of random sequences for deriving encryption keys, multicast group management...

4.4 Convergence Layer

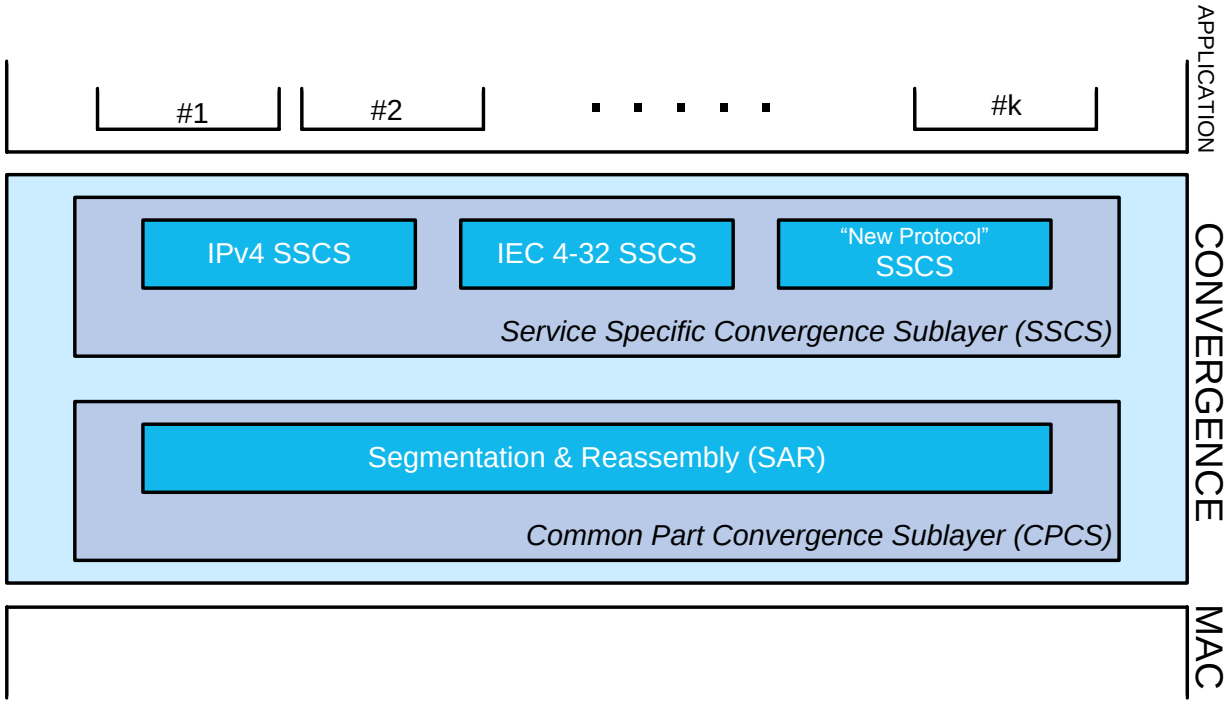
The Convergence Layer (CL) classifies traffic associating it with its proper MAC connection. This layer performs the mapping of any kind of traffic to be properly included in MAC SDUs, providing access to the core MAC functionalities of system access, bandwidth allocation, connection management and mesh topology resolution. It may also include payload header suppression functions.

The convergence layer is separated into two sub layers:

- The **Common Part Convergence Sub layer (CPCS)** provides a set of generic services.
- The **Service Specific Convergence Sub layer (SSCS)** contains services that are specific to one application layer.

There are many SSCS, typically one per application, but only one common part, as shown in Figure 4-4:

Figure 4-4. Convergence Layer



Several convergence sub layers are defined in order to accommodate different kinds of traffic into MAC SDUs, namely, IP convergence layer as a very useful and universal access to PRIME, and IEC 61334-4-32 as a link towards metering systems.

5. Processor and Architecture

5.1 ADD8051C3A Microcontroller Description

The following document provides a detailed description of the ADD8051C3A architecture and its hardware. ADD8051C3A is fully compatible with any 8051 legacy microcontroller, however there are some hardware differences that deserve to be taken into account.

CPU speed has been improved. The machine cycle has been reduced from 12 to 3 clock cycles, and all the instructions are executed in 1 or 2 machine cycles. Programs are executed a minimum of four times faster than in a standard-8051-architecture device, and up to a five times speedup is achieved for most programs.

The microcontroller is intended to work with an up to 256Kbytes external SRAM device. Program memory and XDATA memory share this external SRAM, where lower addresses are always dedicated to store program memory, since upper ones are configurable to allocate different XDATA memory size configurations depending on bank switching page size and SRAM total size.

The size of the extended program pages can be configured to 32K/16K/8K/0K.

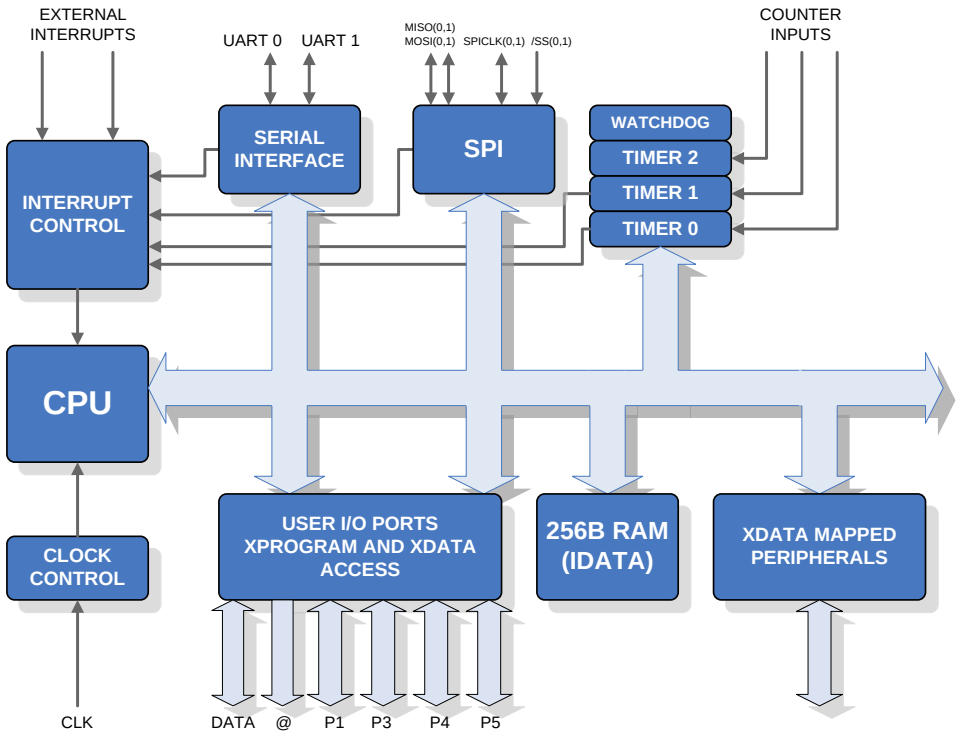
Furthermore, ATPL210 has developed a “compact” memory configuration using a 32Kbytes on-chip SRAM to satisfy applications that use a reduced program code version and in which saving external-circuitry costs and reducing PCB size are prioritary requirements. This configuration can be selected by means of EXTRAM external input:

- EXTRAM=’0’, 32Kbytes On-chip SRAM selected.
- EXTRAM=’1’, External memory device selected. On-chip SRAM is not accessible in this mode.

(Unless otherwise stated, External memory device size =256Kbytes is assumed in this user manual).

This document includes a detailed description of registers and peripherals. Some of these items are upgraded versions of the original 8051 microcontroller circuitry, while others are a simplified version..

Figure 5-1. ADD8051C3A microcontroller block diagram



5.2 Core Pinout Description

Figure 5-2. ADD8051C3A core pinout diagram

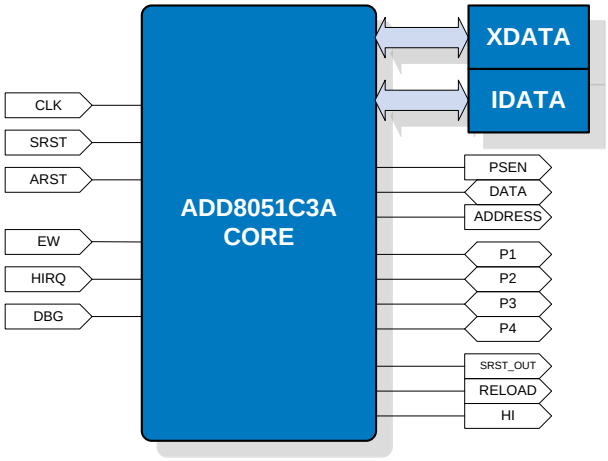


Table 5-1. Core pinout description

Pin	I/O	Description
CLK	I	Clock input.
ARST	I	Asynchronous reset.
SRST	I	SRST Must be held active one clock cycle to ensure proper power-on. External signal D_INIT is connected to this core input.
/EW	I	Watchdog enable Low level active. External signal/EWDG is connected to this core input.
HIRQ	I	Hard idle request High level active. When asserted, the microcontroller finishes the instruction in execution and goes to a special hard idle state. In hard idle state the CPU is stopped while peripheral circuits still run, CPU restarts only after HIRQ deassertion.
DBG	I	Debug mode enable High level active.
PSEN	O	Program Store Enable Read strobe to program memory.
DATA	I/O	Data bus 16-bit bidirectional port to access program and/or data memories.
ADDRESS	O	Address bus Output port to access program memory and XDATA.
SRST_OUT	O	Watchdog timer reset signal.
RELOAD	O	Program reload control signal Software or watchdog activated. Force program reload from a serial storage device to a parallel SRAM.
HI	O	Hard idle flag, This flag is set to ‘1’ when the CPU is in hard idle state.

PORTS: P1, P3, P4, P5 (*)	I/O	I/O Ports. 8-bit pseudo bidirectional I/O ports with pull-up resistors. These ports are by default in **pseudo-bidirectional** configuration. Configured in this way, port pins that have 1s written to them are pulled high by the internal pull-ups, and in that state can be used as inputs. When a bit in a Port register has a 0 to 1 transition the related pin is driven high using transistor during 1 clock cycle, then the transistor is switched off and the pull-up resistor keeps the logic level. Ports P3, P4 and P5 can be also configured as **push-pull** mode ports. In push-pull mode the pin is always in output mode and logic 1 is always high driven (see 5.7.2). Some pins of P3, P4 and P5 also serve the functions of various special features of the microcontroller: P3.0 → RxD (serial port 0 input) P3.1 → TxD (serial port 0 output) P3.2 → INT0 (external interrupt 0) P3.3 → INT1 (external interrupt 1) P3.4 → T0 (timer 0 external input) P3.5 → T1 (timer 1 external input) P3.6 → WR(external data memory write strobe) P3.7 → RD (external data memory read strobe) P4.0 → RxD (serial port 2 input) P4.1 → TxD (serial port 2 output) P4.2 → SS1 (slave select input) P4.3 → SPICLK1 (SPI1 clock input/output) P4.4 → MOSI1 (master out / slave in data) P4.5 → MISO1 (master in / slave out data P4.6 → T2 (timer 2 input/output) P4.7 → T2EX (timer 2 external input) P5.0 → SS0 (slave select input) P5.1 → SPICLK0 (SPI0 clock input/output) P5.2 → MOSI0 (master out / slave in data) P5.3 → MISO0 (master in / slave out data) P5.4 → RxD (serial port 1 input) P5.5 → TxD (serial port 1 output) (*) each bit of ports 1, 3, 4 and 5 is composed of three lines: data output, data input and output enable (low level active)

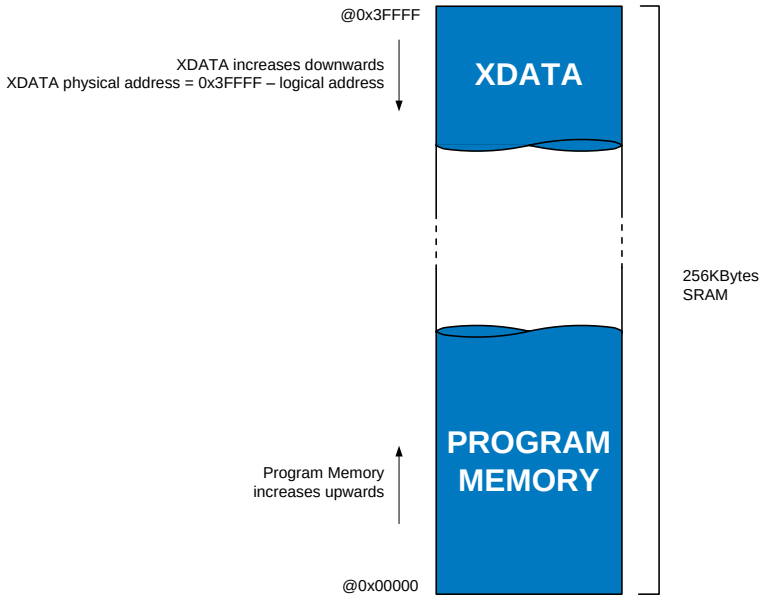
5.3 Memory Organization

ADD8051C3A has separate logical address spaces for program and data memory, as shown in Figure 5-3. The logical separation of program and data memory allows the data memory to be accessed by 8-bit addresses, which can be quickly stored and manipulated by an 8-bit CPU. Nevertheless, 16-bit data memory addresses can also be generated through the DPTR (Data Pointer Registers).

Program memory must only be read. Writing in Program Memory could lead to unexpected results and must be avoided. There is a block of 64kbytes of base program memory, which can be increased by extended program pages.

XDATA occupies a separate address space from Program Memory. In the ADD8051C3A, the lowest 256 bytes of data memory (IDATA) are on chip. From the microcontroller point of view, up to 64Kbytes of XDATA can be addressed in the Data Memory space (ATPL210A allows to address more than 64Kbytes in data memory, see 11). The CPU generates the internal read and write signals, RD and WR, needed during XDATA Memory accesses. In ATPL210A, Program Memory and XDATA Memory are combined in a single SRAM and accessed by the same bus.

Figure 5-3. 256KB SRAM, XDATA and Program Memory



The program must be uploaded to the SRAM after power up. The source of program must be a non volatile storage unit (i.e. serial flash). A specific module (boot loader peripheral) has been designed to control the booting of the system after power up. The boot loader module can also be used to program the non volatile storage unit (serial flash) using a serial link. The non volatile storage unit must be connected to P5 port pins used for the SPI0 link (see Table 5-1). The serial link to the boot loader shares pins with the microcontroller Serial Port 0. For more information see 10.

5.3.2 Program Memory

After reset, the CPU begins execution from location 0000H and stack pointer (SP) value is set to 07H.

Interrupt Handling

A fixed location in Program Memory is assigned to each interrupt. The interrupt causes the CPU to jump to that location, where it begins executing the service routine.

Example: External Interrupt 0, is assigned to location 0003H. If External Interrupt 0 is going to be used, its service routine must begin at location 0003H. If the interrupt is not going to be used, its service location is available as general purpose Program Memory.

The interrupt service locations are spaced at 8-byte intervals beginning at 0x0003 (i.e.0x0003 for External Interrupt0, 0x000B for Timer0, etc.).

If an interrupt service routine is short enough, it can reside entirely within that 8-byte interval. Longer service routines can use a jump instruction to skip over subsequent interrupt locations, if other interrupts are in use.

Program Memory addresses are always 17 bits wide, even though the actual amount of Program Memory used may be less than 64kbytes.

P0 and P2 ports are no longer used to access external memories, but Special Function Registers (SFR) P0 and P2 are still functional.

- P2 register is used to generate the high order address byte when executing MOVX A,@Ri or MOVX @Ri,A.
- P0 register is the program address control register and it is used to specify the size of extended program pages and to control bank switching

5.3.3 Extended Addressing

ATPL210A combines program code and data in a single SRAM.

ADDRESS to access program is generated as a function of P0(7:6), P0(5:0) and PC(15:0).

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
P0	SX1	SX0	P05	P04	P03	P02	P01	P00

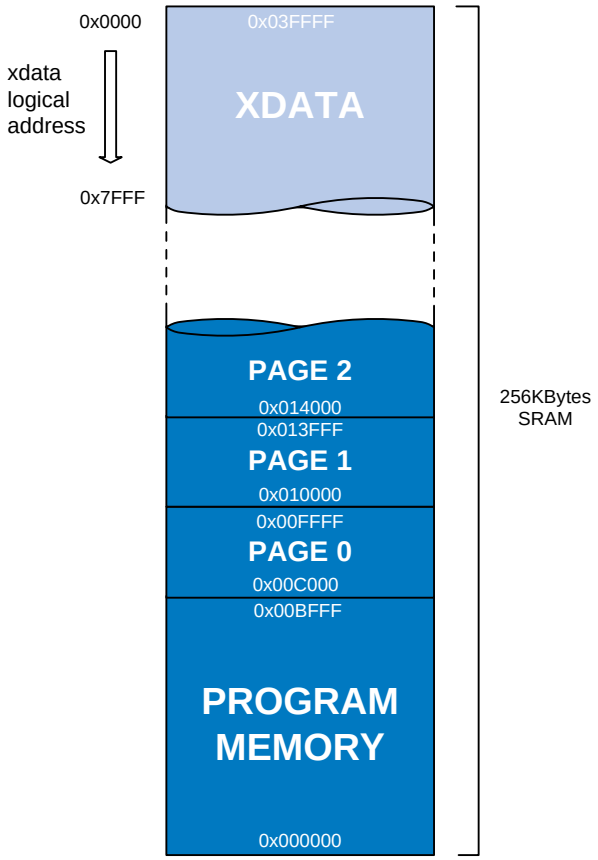
- SX(1:0): Size of extended program pages and common program space

SX(1:0)	Extended	Common
“11”	0KB	64KB
“10”	8KB	56KB
“01”	16KB	48KB
“00”	32KB	32KB

- P0(5:0): Extended program page number

Allowing different configurations as the one shown in Figure 5-4

Figure 5-4. SRAM Extended Addressing. 48KB common memory + 3x16KB extended program pages



Common and extended memory sizes vary depending on the value in P0(7:6), as shown above. According to these size values, the core knows when the PC is pointing to an address located in common memory and when it is pointing to an address located in extended memory. When PC is pointing to an address in extended memory, then P0(5:0) indicates the extended page number and the physical address is automatically calculated. Some examples are shown below.

Example: SX(1:0)=”00” (common memory size = 32KB)

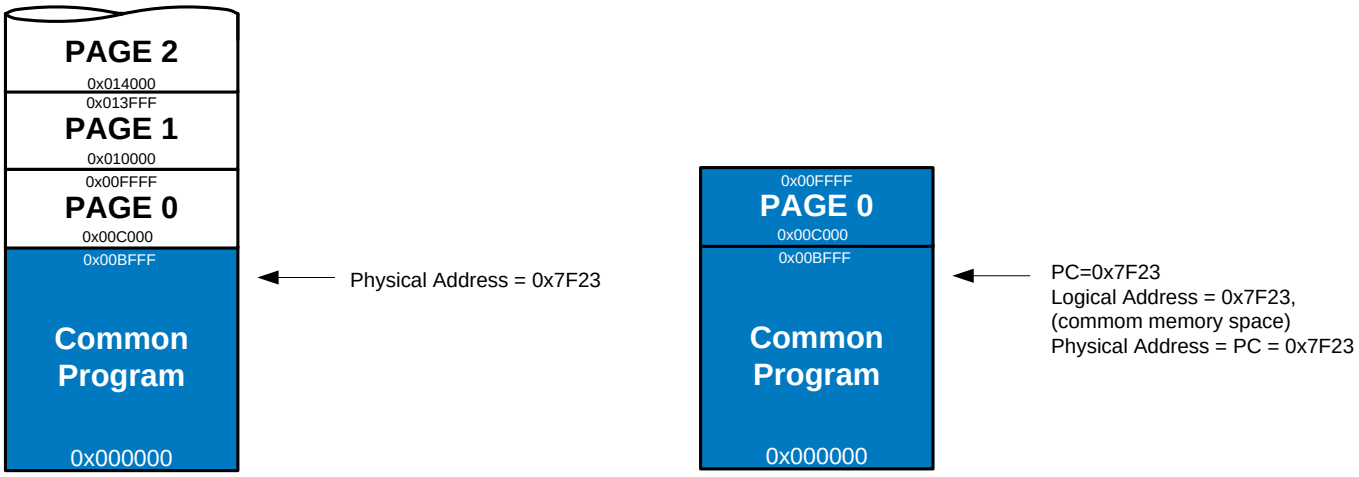
PC=0x7F23 (PC pointing to an address in common memory space)

P0(5:0)= don’t care

LOGIC ADDRESS=0x7F23, common memory space

PHYSICAL ADDRESS=0x7F23 (common memory space → P0(5:0) is ignored)

Figure 5-5. Extended Addressing example 1



Example: SX(1:0)=”00” (common memory size = 32KB)

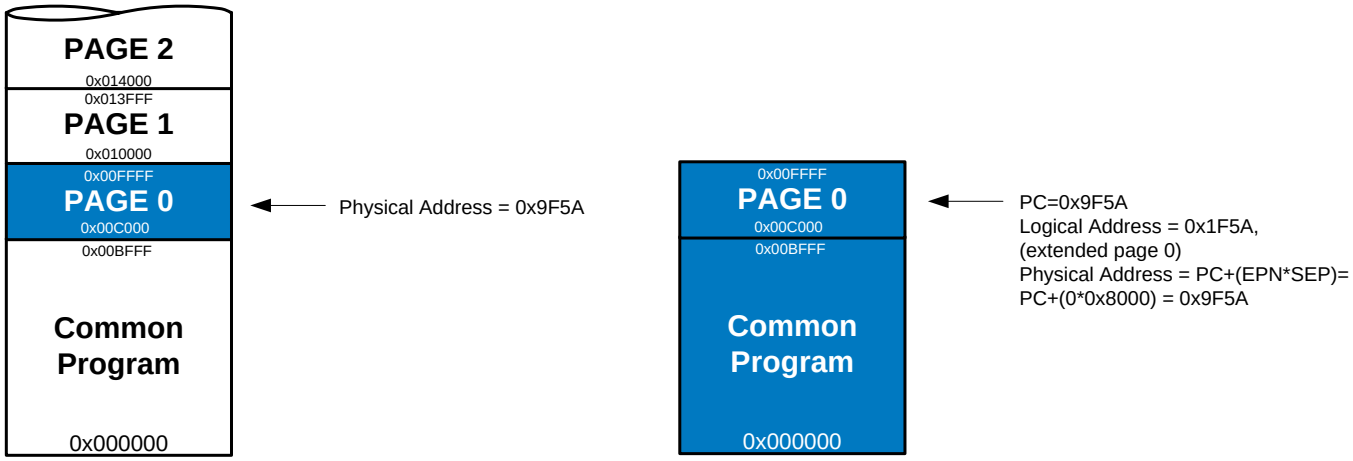
PC=0x9F5A (PC pointing to an address in extended memory)

P0(5:0)= 0x00 (extended page 0)

LOGIC ADDRESS=0x1F5A, extended page 0

PHYSICAL ADDRESS=0x9F5A

Figure 5-6. Extended Addressing example 2



Example: SX(1:0)=”00” (common memory size = 32KB)

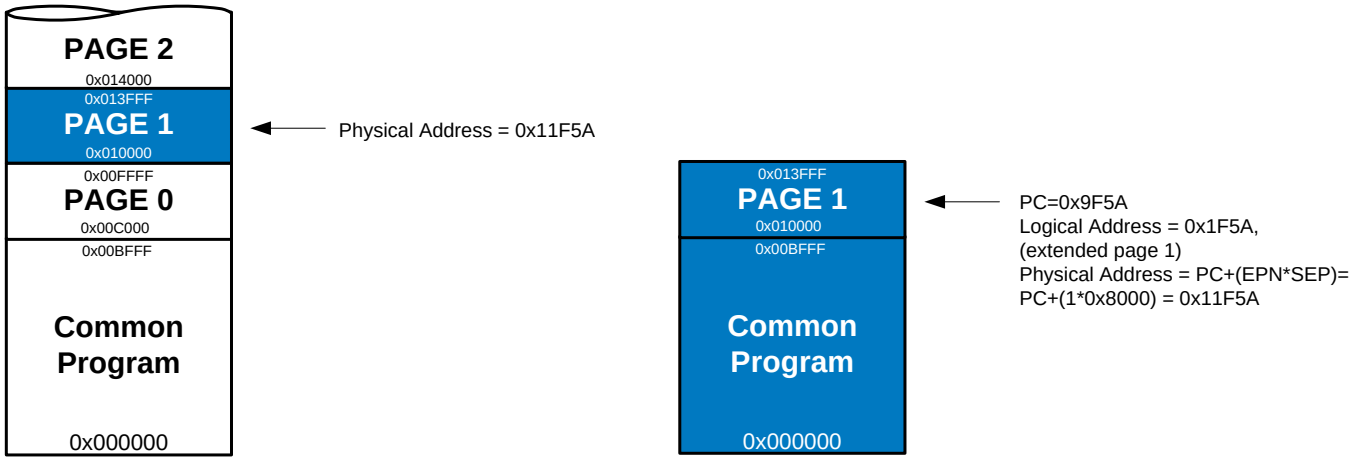
PC=0x9F5A (PC pointing to an address in extended memory)

P0(5:0)= 0x01 (extended page 1)

LOGIC ADDRESS=0x1F5A, extended page 1

PHYSICAL ADDRESS=0x11F5A

Figure 5-7. Extended Addressing example 3



Thus if PC < SCS then Physical Address = PC

else Physical Address = PC+(EPN*SEP)

where SCS= Size of Common Space;

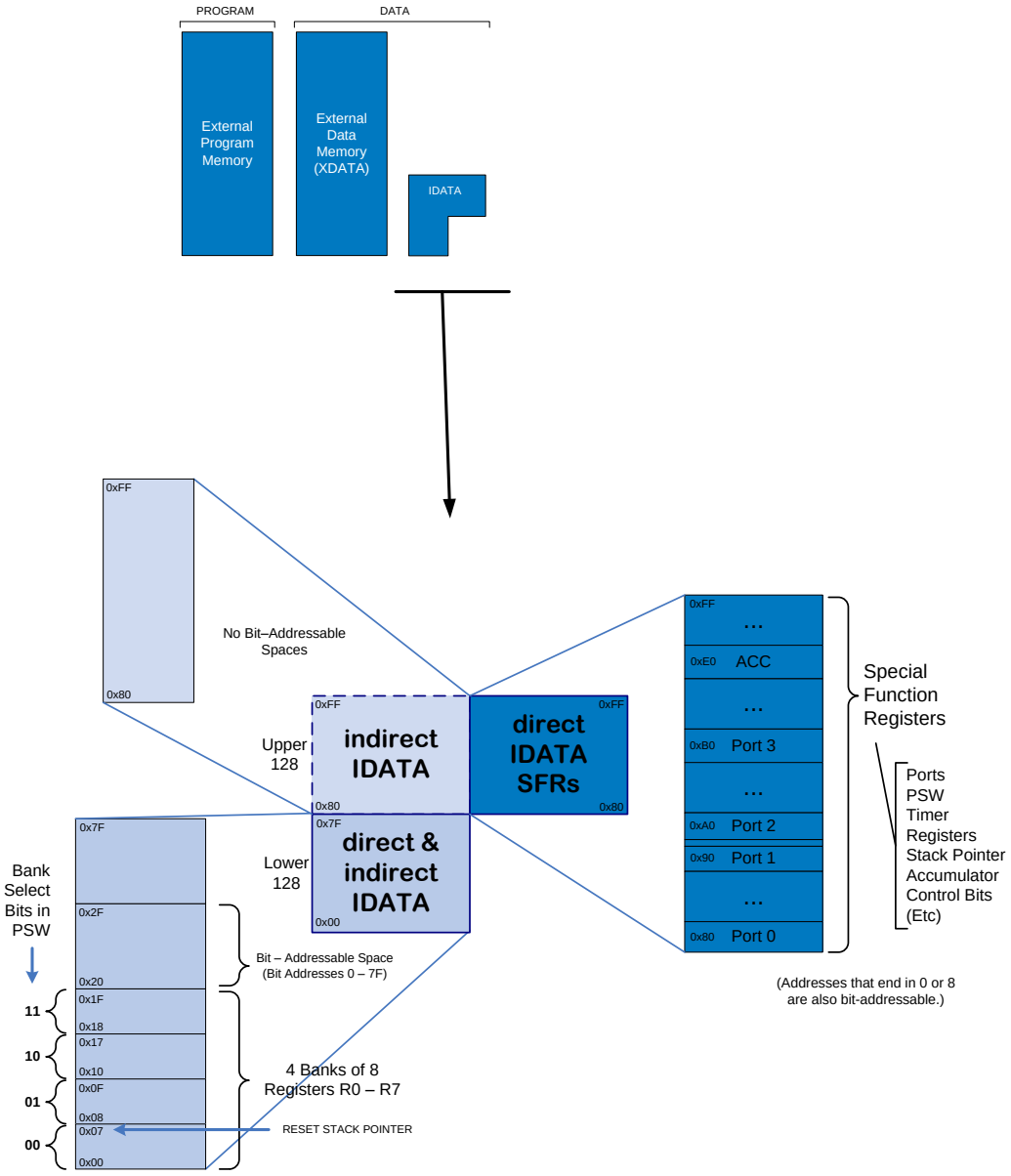
EPN=Extended Page Number;

SEP= Size of Extended Pages(defined by SX(1:0) value).

5.3.4 Data Memory

Figure 5-8 shows the internal and external Logical Data Memory spaces available to the microcontroller user.

Figure 5-8. Data Memory Space



Logical addressing of the Internal Data Memory is mapped as shown in Figure 5-8**.** The internal IDATA memory space has a total size of 384 bytes and is divided into three blocks, which are generally referred to as the Lower 128 bytes, the Upper 128 bytes and SFR space (128 bytes size).

Internal Data Memory addresses are always one byte wide, which implies an address space of only 256 bytes. A simple trick is used to accommodate the 384 bytes of IDATA using 8 bit addresses: direct addresses higher than 7FH access SFR memory space, whereas indirect addresses higher than 7FH access the Upper 128 bytes of IDATA. Thus, Figure 5-8 shows the Upper 128 bytes and SFR space occupying the same block of addresses, 80H through FFH, although they are physically separate entities.

In the Lower 128 bytes of IDATA, the lowest 32 bytes are grouped into 4 banks of 8 registers. Program instructions call out these registers as R0 through R7. Two bits in the Program Status Word (PSW) select which register bank is in use. This allows more efficient use of code space, since register instructions are shorter than instructions that use direct addressing.

The next 16 bytes above the register banks form a block of bit-addressable memory space. The 80C51 instruction set includes a wide selection of single-bit instructions, and the 128 bits in this area can be directly addressed by these instructions. The bit addresses in this area are 00H through 7FH.

Summarizing, the Lower 128 bytes are accessed independently of the addressing mode (direct or indirect addressing). The Upper 128 bytes can be accessed only by indirect addressing, whereas SRF space is accessed only by direct addressing.

Table 5-2 gives a look at the Special Function Register (SFR) space. SFRs include the Port registers, timers, peripheral controls, etc. These registers can only be accessed by direct addressing. Sixteen addresses in SFR space are both byte and bit addressable. The bit addressable SFRs are those whose address ends in 0H or 8H. ADD8051C3A includes the same 21 SFRs included in the legacy 8051C plus additional SFRs to implement the extended features.

The CPU generates the RD (P3.7) and WR (P3.6) signals needed during SRAM accesses, as shown below.

5.3.5 SFR Registers

Table 5-2 shows a map of the internal memory area called the Special Function Register (SFR) space. The entries in bold are registers not included in the standard architecture of the 8051, these entries are specific for the ADD8051C3A. Blank entries are not implemented on the chip. User software should not write to these unimplemented locations. Read accesses to these addresses will return random data.

Table 5-2. Special Function Registers (specific ADD8051C3A registers in bold)

F8								**T3**	FF
F0	B	**CONF**							F7
E8									EF
E0	ACC	**SPSTAT**	**SPCTL**	**SPDAT**	**SPSTAT1**	**SPCTL1**			E7
D8	**SPDAT10**	**SPDAT11**	**SPDAT12**	**SPDAT13**	**SPDAT14**	**SPDAT15**	**SPDAT16**	**SPDAT17**	DF
D0	PSW								D7
C8	**T2CON**	**T2MOD**	**RCAP2L**	**RCAP2H**	**TL2**	**TH2**			CF
C0	**P4**				**P5**				C7
B8	IP								BF
B0	P3							**IPH**	B7
A8	IE								AF
A0	P2	**IE2**	**AUX1**	**AUX2**					A7
98	SCON	SBUF	**SCON1**	**SBUF1**	**P3M**	**P4M**	**P5M**		9F
90	P1	**IP2HL**					**T1NP**		97
88	TCON	TMOD	TL0	TL1	TH0	TH1	**TPR**	**T1NC**	8F
80	P0	SP	DPL	DPH				PCON	87

The functions of some SFRs are described in the text below, while others are described with their related peripherals.

- **Accumulator (address: E0H**)

  ACC is the Accumulator register. Note that mnemonics for accumulator specific instructions usually refer to the Accumulator simply as A.

- **B Register (address: F0H)**

  The B register is used during multiply and divide operations. For other instructions it can be treated as another scratch pad register.

- **PSW register (address: D0H)**

  The PSW register contains program status information as detailed below.

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
PSW	CY	AC	F0	RS1	RS0	OV	F1	P

  - CY: Carry flag
  - AC: Auxiliary carry flag
  - F0: General purpose flag
  - RS(1:0): Register bank select control bits
  - OV: Overflow flag
  - F1: User definable flag
  - P: Parity flag

- **Stack Pointer (address: 81H)**

  8 bit stack pointer that is initialized to internal memory of 07H. The user program can initialize it at any internal RAM location ranging from 07H to FFH.

- **Auxiliary 1**

  The AUX1 register includes the data pointer selection bit, the software reset control and the switch to enable wake-up from power-down using external interrupts.

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
AUX1	--	--	SRST	GF2	WUPD	0	--	DPS

  - --: Reserved bits
  - SRST: Software reset
  - GF2: General purpose flag
  - WUPD: When set, enables external interrupts driven wake-up from power down
  - 0: fixed ‘0’
  - DPS: Data pointer selector, selects between DPTR0 and DPTR1

- **P0, P1, P2, P3, P4, P5 registers**

  P1, P3, P4 and P5 are the SFR registers of Ports 1, 3, 4 and 5 respectively.

  Writing a one/zero to a bit in these registers causes the corresponding port output pin to switch high/low. When a port bit is used as an input the corresponding port SFR must be set to ‘1’.

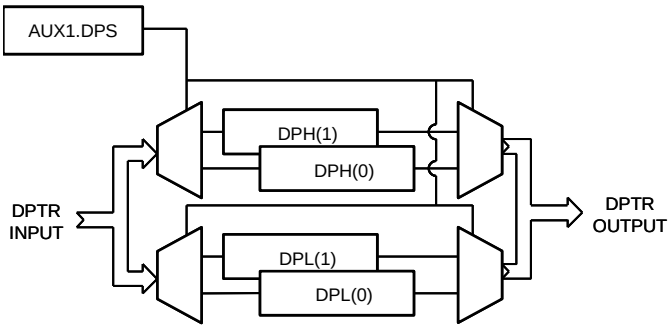
  Ports 3, 4 and 5 are pseudo bidirectional by default, and can be configured to push-pull or pseudo bidirectional using SFRs P3M, P4M and P5M respectively, as follows: when P4M(i) is set, P4(i) is configured as push-pull.

P0 and P2 ports are no longer used to address program and external data. P0 SFR takes control of extended addressing and program banking and P2 SFR is used when MOVX @Ri instructions are executed (in this case, P2 SFR content is used as most significant address byte).

- **Data Pointer**

  The Data Pointer Register (DPTR) consists of a high byte (DPH) and a low byte (DPL). Its intended function is to hold a 16-bit address. It may be manipulated as a 16-bit register or as two independent 8-bit registers. The same entry accesses two different data pointers (DPTR0, DPTR1), user software can switch between them using the data pointer selection bit in AUX1.

Figure 5-9. DPTR selection scheme



- **Serial Data Buffers**

  The Serial Buffers have actually two separate registers, a transmit buffer and a receive buffer. When data is moved to SBUF, it goes to the transmit buffer and is held for serial transmission. (Moving a byte to SBUF is what initiates the transmission.) When data is moved from SBUF, it comes from the receive buffer.

- **Timer Registers**

  Register pairs (TH0, TL0), (TH1, TL1) and (TH2, TL2) are the 16-bit Counting registers for Timer/Counters 0, 1 and 2, respectively.

  T3 is the 8-bit counting register of the watchdog timer, see 6.3 for detailed behavior description.

- **CONF Register**

  CONF register includes specific configuration features for the flash loader and Standard Serial Interface.

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
CONF	--	--	--	--	--	RLD	RLT3	BAUD2

  - --: Reserved bits
  - RLD: When set, it forces program reload from SPI flash
  - RLT3: When set, it forces program reload from SPI flash when watchdog timeout occurs
  - BAUD2: doubles USART baud rates when set

- **Control Registers**

  Special Function Registers IP, IPH, IP2HL, IE, IE2, TMOD, TCON, T2CON, SCONj, SPSTATj and PCON contain control and status bits for the interrupt system, the Timer/Counters, the serial ports and the serial peripheral interfaces

5.4 Instruction Set

The instruction set provides a variety of fast addressing modes for accessing the internal RAM to facilitate byte operations on small data structures. The instruction set provides extensive support for one-bit variables as a separate data type, allowing direct bit manipulation in control and logic systems that require boolean processing.

5.4.1 Program Status Word

The Program Status Word (PSW) contains several status bits that reflect the current state of the CPU. The PSW, shown in 5.3.5, resides in the SFR space. It contains the Carry bit, the Auxiliary Carry (for BCD operations), the two register bank select bits, the Overflow flag, a Parity bit, and two user-definable status flags.

The Carry bit, other than serving the function of a Carry bit in arithmetic operations, also serves as the “Accumulator” for a number in Boolean operations.

The bits RS0 and RS1 are used to select one of the four register banks shown in Figure 5-8. A number of instructions refer to these RAM locations as R0 through R7. The selection of which of the four is being referred to is made on the basis of the RS0 and RS1 at execution time.

The Parity bit reflects the number of 1s in the Accumulator: P = 1 if the Accumulator contains an odd number of 1s, and P = 0 if the Accumulator contains an even number of 1s. Thus the number of 1s in the Accumulator plus P is always even. Two bits in the PSW are uncommitted and may be used as general purpose status flags.

5.4.2 Addressing Modes

- Direct Addressing

  In direct addressing the operand is specified by an 8-bit address field in the instruction. Only internal Data RAM and SFRs can be directly addressed.

- Indirect Addressing

  In indirect addressing the instruction specifies a register which contains the address of the operand. Both internal and external RAM can be indirectly addressed.

  The address register for 8-bit addresses can be either R0 / R1 of the selected register bank or the Stack Pointer. The address register for 16-bit addresses can only be the 16-bit “data pointer” register, DPTR.

- Register Instructions

  The register banks, containing registers R0 through R7, can be accessed by certain instructions which carry a 3-bit register specification within the opcode of the instruction. Instructions that access the registers this way are code efficient, since this mode eliminates an address byte. When the instruction is executed, one of the eight registers in the selected bank is accessed. One of four banks is selected at execution time by the two bank select bits in the PSW.

- Register-Specific Instructions

  Some instructions are specific to a certain register. For example, some instructions always operate on the Accumulator, or Data Pointer, etc., so no address byte is needed to point to it. The opcode itself does that. Instructions that refer to the Accumulator as A assemble as accumulator specific opcodes.

- Immediate Constants

  The value of a constant can follow the opcode in Program Memory.

  For example:

  MOV A, 64H

  loads the Accumulator with the number 64H.

- Indexed Addressing

  Only program Memory can be accessed with indexed addressing, and it can only be read. This addressing mode is intended for reading look-up tables in Program Memory. A 16-bit base register (either DPTR or the Program Counter) points to the base of the table, and the Accumulator is set up with the table entry number.

The address of the table entry in Program Memory is formed by adding the Accumulator data to the base pointer.

Another type of indexed addressing is used in the “case jump” instruction. In this case the destination address of a jump instruction is computed as the sum of the base pointer and the Accumulator data.

5.4.3 Arithmetic Instructions

The menu of arithmetic instructions is listed in Table 5-3, which indicates the addressing modes that can be used with each instruction to access the <byte> operand. For example, the ADD A,<byte> instruction can be written as:

ADD A, 6EH (direct addressing)

ADD A, @R1 (indirect addressing)

ADD A, R6 (register addressing)

ADD A, #206 (immediate constant)

The execution times listed in Table 5-3 assume 3 clock cycles per machine cycle (mc), using a 10MHz clock 1 mc is executed in 0.3us. All the arithmetic instructions execute in 1mc except the Multiply and Divide instructions, which take 2mc.

Note that any byte in the internal Data Memory space can be incremented without going through the Accumulator.

One of the INC instructions operates on the 16-bit Data Pointer. The Data Pointer is used to generate 16-bit addresses for external memory, so being able to increment it in one 16-bit operation is a useful feature.

The MUL AB instruction multiplies the Accumulator by the data in the B register and puts the 16-bit product into the concatenated B and Accumulator registers.

The DIV AB instruction divides the Accumulator by the data in the B register and leaves the 8-bit quotient in the Accumulator, and the 8-bit remainder in the B register. Oddly enough, DIV AB finds less use in arithmetic “divide” routines than in radix conversions and programmable shift operations. In shift operations, dividing a number by 2^n shifts its n bits to the right. Using DIV AB to perform the division completes the shift in 2mc and leaves the B register holding the bits that were shifted out.

The DA A instruction is for BCD arithmetic operations. In BCD arithmetic, ADD and ADDC instructions should always be followed by a DA A operation, to ensure that the result is also in BCD. Note that DA A will not convert a binary number to BCD. The DA A operation produces a meaningful result only as the second step in the addition of two BCD bytes.

Table 5-3. Arithmetic Instructions

MNEMONIC	OPERATION	ADDRESSING MODES				EXECUTION
		DIR	IND	REG	IMM	TIMES (mc)
ADD A,<byte>	A = A + <byte>	X	X	X	X	1
ADD A,<byte>	A = A + <byte> + C	X	X	X	X	1
SUBB A,<byte>	A = A – <byte> – C	X	X	X	X	1
INC A	A = A + 1	Accumulator only				1
INC<byte>	<byte> = <byte> + 1	X	X	X		1
INC DPTR	DPTR = DPTR + 1	Data Pointer only				1
DEC A	A = A – 1	Accumulator only				1
DEC<byte>	<byte> = <byte> – 1	X	X	X		1
MUL AB	B*A = B x A	ACC and B only				2
DIV AB	A = Int[A/B]	ACC and B only				2
	B = Mod[A/B]					
DA A	Decimal Adjust	Accumulator only				1

5.4.4 Logical Instructions

Table 5-4 shows the list of logical instructions. The instructions that perform Boolean operations (AND, OR, Exclusive OR, CPL) on bytes perform the operation on a bit-by-bit basis. That is, if the Accumulator contains 00110101B and byte contains 01010011B, then:

ANL A, <byte>

will leave the Accumulator holding 00010001B.

The addressing modes that can be used to access the <byte> operand are listed in Table 5-4.

The ANL A, <byte> instruction may take any of the forms:

```
ANL   A,6FH     (direct addressing)
ANL   A,@R0     (indirect addressing)
ANL   A,R5      (register addressing)
ANL   A,#42H    (immediate constant)
```

All of the logical instructions execute in 1mc.

Table 5-4. Logical Instructions

MNEMONIC	OPERATION	ADDRESSING MODES				EXECUTION
		DIR	IND	REG	IMM	TIME (mc)
ANL A,<byte>	A = A.AND. <byte>	X	X	X	X	1
ANL <byte>,A	<byte> = <byte> .AND.A	X				1
ANL <byte>,#data	<byte> = <byte> .AND.#data	X				1
ORL A,<byte>	A = A.OR.<byte>	X	X	X	X	1
ORL <byte>,A	<byte> = <byte> .OR.A	X				1
ORL <byte>,#data	<byte> = <byte> .OR.#data	X				1
XRL A,<byte>	A = A.XOR. <byte>	X	X	X	X	1
XRL <byte>,A	<byte> = <byte> .XOR.A	X				1
XRL <byte>,#data	<byte> = <byte> .XOR.#data	X				1
CLR A	A = 00H	Accumulator only				1
CPL A	A = .NOT.A	Accumulator only				1
RL A	Rotate ACC Left 1 bit	Accumulator only				1
RLC A	Rotate Left through Carry	Accumulator only				1
RR A	Rotate ACC Right 1 bit	Accumulator only				1
RRC A	Rotate Right through Carry	Accumulator only				1
SWAP A	Swap Nibbles in A	Accumulator only				1

Note that Boolean operations can be performed on any byte in the internal Data Memory space without going through the Accumulator. The *XRL <byte>, #data instruction*, for example, offers a quick and easy way to invert port bits, as in *XRL P1, #0FFH*.

If the operation is in response to an interrupt, not using the Accumulator saves time and effort to push it onto the stack in the service routine.

The Rotate instructions (RL, A, RLC A, etc.) shift the Accumulator 1 bit to the left or right. For a left rotation, the MSB rolls into the LSB position. For a right rotation, the LSB rolls into the MSB position.

The SWAP A instruction interchanges the high and low nibbles within the Accumulator. This is a useful operation in BCD manipulations.

5.4.5 Data Transfer Instructions

Internal RAM

Table 5-5 shows the menu of instructions that are available for moving data around within the internal memory spaces, and the addressing modes that can be used with each one. All of these instructions are executed in 1 mc.

Table 5-5. Data Transfer Instructions that Access Internal Data Memory Space

MNEMONIC	OPERATION	ADDRESSING MODES				EXECUTION
		DIR	IND	REG	IMM	TIME (mc)
MOV A,<src>	A = <src>	X	X	X	X	1
MOV <dest>,A	<dest> = A	X	X	X		1
MOV <dest>,<src>	<dest> = <src>	X	X	X	X	1
MOV DPTR,#data16	DPTR = 16-bit immediate constant				X	1
PUSH <src>	INC SP:MOV“@SP”,<src>	X				1
POP <dest>	MOV <dest>,“@SP”:DEC SP	X				1
XCH A,<byte>	ACC and <byte> exchange data	X	X	X		1
XCHD A,@Ri	ACC and @Ri exchange low nibbles		X			1

The MOV <dest>, <src> instruction allows data to be transferred between any two internal RAM or SFR locations without going through the Accumulator. Remember, the upper 128 bytes of IDATA can be accessed only by indirect addressing, and SFR space only by direct addressing.

Note that in any 80C51 device, the stack resides in on-chip RAM, and grows upwards. The PUSH instruction first increments the Stack Pointer (SP), then copies the byte into the stack. PUSH and POP use only direct addressing to identify the byte being saved or restored, but the stack itself is accessed by indirect addressing using the SP register. This means the stack can go into the Upper 128 bytes of RAM, if they are implemented, but not into SFR space.

The Data Transfer instructions include a 16-bit MOV that can be used to initialize the Data Pointer (DPTR) for look-up tables in Program Memory, or for 16-bit external Data Memory accesses.

The XCH A, <byte> instruction causes the Accumulator and addressed byte to exchange data. The XCHD A, @Ri instruction is similar, but only the low nibbles are involved in the exchange.

External RAM (from microcontroller point of view)

Table 5-6 shows a list of the Data Transfer instructions that access external Data Memory. Only indirect addressing can be used. The choice is whether to use a one-byte address, @Ri, where Ri can be either R0 or R1 of the selected register bank, or a two-byte address, @DPTR. All of these instructions execute in 2 mc.

Note that in all external Data RAM accesses, the Accumulator is always either the destination or source of the data.

The read and write strobes to external RAM are activated only during the execution of a MOVX instruction.

Table 5-6. Data Transfer Instructions that access External Data Memory Space

ADDRESS WIDTH	MNEMONIC	OPERATION	EXECUTION TIME (mc)
8 bits	**MOVX A,@Ri**	Read external RAM @Ri	2
8 bits	**MOVX @Ri,A**	Write external RAM @ Ri	2
16 bits	**MOVX A,@DPTR**	Read external RAM @ DPTR	2
16 bits	**MOVX @DPTR,A**	Write external RAM @ DPTR	2

Code Constants

Table 5-7 shows the two instructions that are available for reading code constants in Program Memory. Since these instructions access only Program Memory, the code constants can only be read, not updated.

The mnemonic is MOVC for “move constant”. The first MOVC instruction in Table 5-7 can accommodate a table of up to 256 entries numbered 0 through 255. The number of the desired entry is loaded into the Accumulator, and the Data Pointer is set up to point to the beginning of the table. Then, *MOVC A,@A+DPTR* copies the desired table entry into the Accumulator.

The other MOVC instruction works the same way, excepting that the Program Counter (PC) is used as the base address: *MOVC A,@A+PC*

Table 5-7. Code Constants Read Instructions

MNEMONIC	OPERATION	EXECUTION TIME (mc)
MOVC A,@A+DPTR	Read program memory at (A + DPTR)	2
MOVC A,@A+PC	Read program memory at (A + PC)	2

5.4.6 Boolean Instructions

80C51 devices contain a complete Boolean (single-bit) processor. The internal RAM contains 128 addressable bits, and the SFR space can support up to 128 addressable bits as well. All of the port lines are bit-addressable, and each one can be treated as a separate single-bit port. The instructions that access these bits are not just conditional branches, but a complete menu of move, set, clear, complement, OR, and AND instructions. These kinds of bit operations are not easily obtained in other architectures with any amount of byte-oriented software. The instruction set for the Boolean processor is shown in Table 5-8. All bit accesses are by direct addressing.

Bit addresses 00H through 7FH are in the Lower 128, and bit addresses 80H through FFH are in SFR space.

The Carry bit in the PSW is used as the single-bit Accumulator of the Boolean processor. Bit instructions that refer to the Carry bit as C assemble as Carry-specific instructions (CLR C, etc.). The Carry bit also has a direct address, since it resides in the PSW register, which is bit-addressable.

Note that the Boolean instruction set includes ANL and ORL operations, but not the XRL (Exclusive OR) operation.

There is a series of bit-test instructions which execute a jump if the addressed bit is set (JC, JB, JBC) or if the addressed bit is not set (JNC, JNB). JBC executes the jump if the addressed bit is set, and also clears the bit. Thus a flag can be tested and cleared in one operation. All the PSW bits are directly addressable, so the Parity bit, or the general purpose flags, for example, are also available to the bit-test instructions.

Table 5-8. Boolean Instructions

MNEMONIC	OPERATION	EXECUTION TIME (mc)
ANL C,bit	C = C.AND.bit	1
ANL C,/bit	C = C.AND..NOT.bit	1
ORL C,bit	C = C.OR.bit	1
ORL C,/bit	C = C.OR..NOT.bit	1
MOV C,bit	C = bit	1
MOV bit,C	bit = C	1
CLR C	C = 0	1
CLR bit	bit = 0	1
SETB C	C = 1	1
SETB bit	bit = 1	1
CPL C	C = .NOT.C	1
CPL bit	bit = .NOT.bit	1
JC rel	Jump if C = 1	2
JNC rel	Jump if C = 0	2
JB bit,rel	Jump if bit = 1	2
JNB bit,rel	Jump if bit = 0	2
JBC bit,rel	Jump if bit = 1; CLR bit	2

Relative Offset

The destination address for these jumps is specified to the assembler by a label or by an actual address in Program memory. However, the destination address assembles to a relative offset byte. This is a signed (two's complement) offset byte which is added to the PC in two's complement arithmetic if the jump is executed. The range of the jump is therefore –128 to +127 Program Memory bytes relative to the first byte following the instruction.

5.4.7 Jump Instructions

Table 5-9 shows the list of unconditional jumps and the execution time associated.

The table lists SJMP, LJMP, and AJMP, which differ in the format of the destination address. JMP is a generic mnemonic which can be used if the programmer does not care which way the jump is encoded.

The SJMP instruction encodes the destination address as a relative offset, as described above. The instruction is 2 bytes long, consisting of the opcode and the relative offset byte. The jump distance is limited to a range of –128 to +127 bytes relative to the instruction following the SJMP.

The LJMP instruction encodes the destination address as a 16-bit constant. The instruction is 3 bytes long, consisting of the opcode and two address bytes. The destination address can be anywhere in the 64k Program Memory space.

The AJMP instruction encodes the destination address as an 11-bit constant. The instruction is 2 bytes long, consisting of the opcode, which itself contains 3 of the 11 address bits, followed by another byte containing the low 8 bits of the destination address. When the instruction is executed, these 11 bits replace the low 11 bits in the PC. The high 5 bits stay the same. Hence the destination has to be within the same 2k block as the instruction following the AJMP.

The JMP @A+DPTR instruction supports case jumps. The destination address is computed at execution time as the sum of the 16-bit DPTR register and the Accumulator. Typically, DPTR is set up with the address of a jump table.

Table 5-9 shows two “CALL addr” instructions LCALL and ACALL, which differ in the format in which the subroutine address is given to the CPU. CALL is a generic mnemonic which can be used if the programmer does not care which way the address is encoded.

The LCALL instruction uses the 16-bit address format, and the subroutine can be anywhere in the 64k Program Memory space.

The ACALL instruction uses the 11-bit format, and the subroutine must be in the same 2k block as the instruction following the ACALL.

Subroutines should end with a RET instruction, which returns execution to the instruction following the CALL.

RETI is used to return from an interrupt service routine. The only difference between RET and RETI is that RETI tells the interrupt control system that the interrupt in progress is done. If there is no interrupt in progress at the time RETI is executed, then the RETI is functionally identical to RET.

Table 5-10 shows the list of conditional jumps available to the microcontroller user. All of these jumps specify the destination address by the relative offset method, and so are limited to a jump distance of –128 to +127 bytes from the instruction following the conditional jump instruction.

There is no Zero bit in the PSW. The JZ and JNZ instructions test the Accumulator data for that condition.

The DJNZ instruction (Decrement and Jump if Not Zero) is for loop control.

The CJNE instruction (Compare and Jump if Not Equal) can also be used for loop control. Two bytes are specified in the operand field of the instruction. The jump is executed only if the two bytes are not equal. Another application of this instruction is in “greater than, less than” comparisons. The two bytes in the operand field are taken as unsigned integers. If the first is less than the second, then the Carry bit is set (1). If the first is greater than or equal to the second, then the Carry bit is cleared.

Table 5-9. Unconditional Jump instructions

MNEMONIC	OPERATION	EXECUTION TIME (mc)
(S)JMP addr	Jump to addr	2
(L)JMP addr	Jump to addr	2
(A)JMP addr	Jump to addr	2
JMP @A+DPTR	Jump to A + DPTR	2
(A,L)CALL addr	Call subroutine at addr	2
RET	Return from subroutine	2
RETI	Return from interrupt	2
NOP	No operation	1

Table 5-10. Conditional Jump instructions

MNEMONIC	OPERATION	ADDRESSING MODES				EXECUTION
		DIR	IND	REG	IMM	TIME(mc)
JZ rel	Jump if A = 0					2
JNZ rel	Jump if A /= 0					2
DJNZ <byte>,rel	Decrement and jump if not zero	X		X		2
CJNE A,<byte>,rel	Jump if A /= <byte>	X			X	2
CJNE <byte>,#data,rel	Jump if <byte> /= #data		X	X		2

5.5 CPU Timing

Execution time is divided into machine cycles. A machine cycle consists of a sequence of 3 states, numbered S1 through S3. Each state time lasts for one clock period. Thus a machine cycle takes 3 clock periods (or near 0.15µs if the clock frequency is 20MHz).

Figure 5-10 shows fetch/execute sequences for various kinds of instructions; while an instruction is being executed code bytes for the next instruction are fetched. Normally three program fetches are generated during each machine cycle, even if the next instruction to be executed doesn't require it. If the instruction doesn't need the three code bytes, the CPU simply ignores the extra fetches, and the Program Fetch Counter is not incremented.

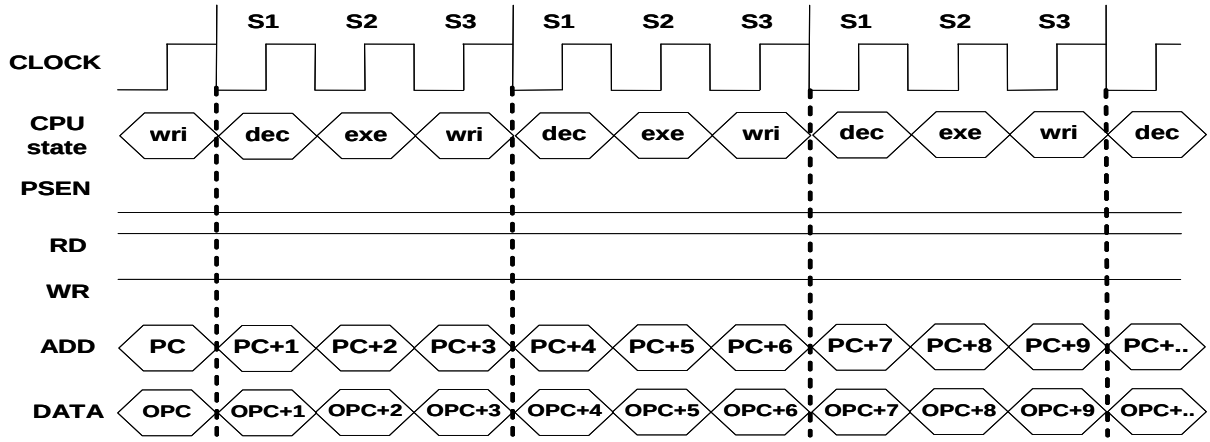
All the instructions, except MOVX, have the same fetch/execute sequence.

The MOVX instructions take two machine cycles to execute. During a MOVX instruction only one fetch is generated, S1 in machine cycle 1. This is the only time program fetches are skipped. The fetch/execute sequence for MOVX instructions is shown in Figure 5-10b and Figure 5-10c.

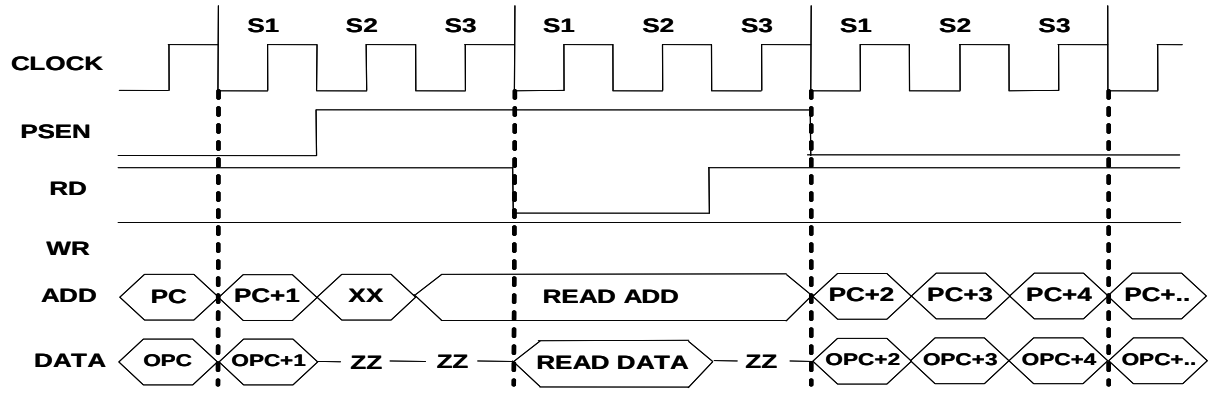
The fetch/execute sequences are the same whether the Program Memory is internal or external to the chip. Execution times do not depend on whether the Program Memory is internal or external.

Figure 5-10 shows the signals and timing involved in program fetches when the Program Memory is external. If Program Memory is external, then the Program Memory read strobe PSEN is normally activated ('0') three times per machine cycle, as shown in Figure 5-10a. If an access to external Data Memory occurs, as shown in Figure 5-10b and 15c, five PSENs are skipped, because the address and data bus are being used for the Data Memory access.

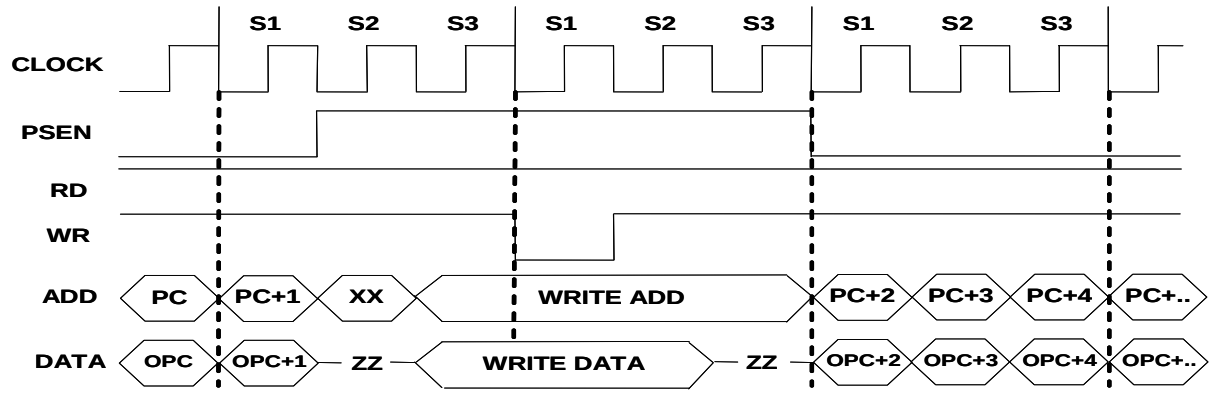
Figure 5-10. Bus timing in 8051C3A



(a) Program execution without MOVX



(b) Program execution with a read MOVX



(c) Program execution with a write MOVX

5.5.2 Reset Modes

ATPL210A can be reset in different ways, listed and explained below

- **External reset.** The external reset input signal is RSTA which is an asynchronous reset. Every time RSTA signal is asserted, the boot loader is triggered (thus, microcontroller is halted, target program from the serial flash is uploaded to the SRAM, and then microcontroller is released to begin program execution). In power-on, D_INIT must be set before RSTA in order to ensure proper system start up.
- **Software reset →** AUX1(5) = AUX1.SRST. This field in AUX1 SFR forces the microcontroller to start program execution from PC=0, and all the SFRs are initialized to their reset default values. This reset doesn't affect the boot loader reload.
- **Software reset →** CONF(2) = CONF.RLD. This field is directly connected to boot loader reload input. Every time the microcontroller sets this field, the boot loader is triggered.
- **Watchdog reset →** /EWDG signal and CONF(1)=CONF.RLT3.
  - CONF.RLT3 = '0' : After a watchdog timeout, microcontroller starts execution from PC=0, but boot loader is not triggered.
  - CONF.RLT3 = '1' **:** After a watchdog timeout, if field RLT3 in CONF SFR is '1', boot loader is triggered.

The microcontroller keeps track of the start-up point in the AUX2 register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
AUX2	--	--	--	--	--	ST2	ST1	ST0

- --: Reserved bits
- ST(2:0): Start-up flags
  - "000":Asynchronous external reset
  - "001":Software reset by writing AUX1(5)='1'
  - "010":Watchdog T3 timeout
  - "100":Asynchronous external reset

SFR values after a reset are shown in Table 5-11. The internal RAM is not affected. Port registers are initialized to FFH to keep port pins in pull-up state.

Table 5-11. SFRs Reset values

SFR	Address	Reset Value
P0	80H	11111111
SP	81H	00000111
DPL	82H	00000000
DPH	83H	00000000
PCON	87H	00000000
TCON	88H	00000000
TMOD	89H	00000000
TL0	8AH	00000000
TL1	8BH	00000000
TH0	8CH	00000000
TH1	8DH	00000000
TPR	8EH	00111111
T1NC	8FH	00000000
P1	90H	11111111
IP2HL	91H	xxxx0000
T1NP	96H	00000000
SCON	98H	00000000
SBUF	99H	00000000
SCON1	9AH	00000000

SFR	Address	Reset Value
SBUF1	9BH	00000000
P3M	9CH	00000000
P4M	9DH	00000000
P5M	9EH	00000000
P2	A0H	11111111
IE2	A1H	xxxxxx00
AUX1	A2H	00000000
AUX2	A3H	xxxxxbbb
IE	A8H	00000000
P3	B0H	11111111
IPH	B7H	00000000
IP	B8H	00000000
P4	C0H	11111111
P5	C4H	11111111
T2CON	C8H	00000000
T2MOD	C9H	xx001100
RCAP2L	CAH	00000000
RCAP2H	CBH	00000000
TL2	CCH	00000000

SFR	Address	Reset Value
TH2	CDH	00000000
PSW	D0H	00000000
SPDAT10	D8H	00000000
SPDAT11	D9H	00000000
SPDAT12	DAH	00000000
SPDAT13	DBH	00000000
SPDAT14	DCH	00000000
SPDAT15	DDH	00000000
SPDAT16	DEH	00000000
SPDAT17	DFH	00000000
ACC	E0H	00000000
SPSTAT	E1H	000xxxxx
SPCTL	E2H	00000100
SPDAT	E3H	00000000
SPSTAT1	E4H	000xx000
SPCTL1	E5H	00000100
B	F0H	00000000
CONF	F1H	00000000
T3	FFH	00000000

5.5.3 Power Saving Modes

ADD8051C3A has two power-saving modes, Idle and Power Down.

In the Idle mode (IDL = 1), the CPU is stopped but the peripheral circuits continue running.

In Power Down (PD = 1), the CPU and the peripheral circuits are stopped, except external interrupts hardware that can be configured to run when in power down mode.

The Idle and Power Down Modes are activated by setting bits in Special Function Register PCON. If 1s are written to PD and IDL at the same time, PD takes precedence. When watchdog is enabled power down modes are automatically disabled.

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
PCON	SMOD	--	--	WLE	GF1	GF0	PD	IDL

- SMOD: Double baud rate bit when timer 1 is used as time generator and serial port is in modes 1, 2 or 3
- --: Reserved bits
- WLE: Watchdog load enable. It must be set by software to enable T3 reload. It is reset by hardware after 13 machine cycles
- GF1: General purpose flag bit, user programmable
- GF0: General purpose flag bit, user programmable
- PD: Sets power-down mode
- IDL: Sets idle mode

5.5.4 Idle Mode

Setting PCON(0), CPU is stopped but peripheral circuits (Interrupts, Timers, Standard Serial Interface and Serial Peripheral Interface functions) continue running. The CPU status is entirely preserved; Stack Pointer, Program Counter, Program Status Word, Accumulator, and all other registers maintain their data during Idle. The port pins hold the logical states they had at the time Idle was activated. PSEN holds at logic high level.

There are two ways to terminate the Idle mode.

- Activation of any enabled interrupt will cause PCON(0) to be cleared by hardware, terminating the Idle mode. The interrupt will be served, and following RETI, the next instruction to be executed will be the one following the instruction that put the device into Idle.
- Reset signal. The signal at the RSTA pin redefines all the SFR values to their reset values and then the IDL bit is cleared. At this time, and with three clock cycle delay, the CPU restart program execution with program counter at initial value 0x0000, RAM content remains unchanged.

5.5.5 Power-Down Mode

An instruction that sets PCON(1) causes that to be the last instruction executed before going into the Power Down mode. In the Power Down mode all functions are stopped, except external interrupts hardware. The contents of the on-chip RAM and Special Function Registers are maintained. The port pins output the values held by their respective SFRs. The PSEN output is held low.

There are two ways to exit Power Down mode.

- Hardware reset. Hardware reset redefines all the SFRs.
- External interrupt. External interrupts must be configured for low level activation.When they are enabled a low level at their inputs will cause a microcontroller restart and a jump to the corresponding interruption routine. As in idle mode, the interrupt will be served, and following RETI, the next instruction to be executed will be the one following the instruction that put the device into power down.

5.6 Interrupts

The microcontroller has 5 user interrupt sources:

- 1 timer interrupt (Timer2).
- 2 serial port interrupts (Standard Serial 0 & Standard Serial 1).
- 2 serial peripheral interface interrupts (SPI0 & SPI1).

And 4 internal system interrupt sources:

- 2 timer interrupts (Timer0 & Timer1)
- 2 generic interrupts (INT0 & INT1)

Interrupt priorities are user selectable to a priority level ranging from “000” to “011” (higher value correspond to higher priority level).

5.6.1 Interrupt Enabling

Each interrupt source can be individually enabled or disabled by setting or clearing a bit in the SFRs named IE and IE2 (Interrupt Enable). The IE register also contains a global disable bit, which can be cleared to disable all interrupts at once.

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
IE	EA	ET2	ESPI	ES	ET1	EX1	ET0	EX0

- EA: ‘1’ enables all interrupts; ‘0’ disables all interrupts
- ET2: ‘1’ enables timer 2 interrupt; ‘0’ disables timer 2 interrupt
- ESPI: ‘1’ enables serial peripheral interface0 interrupt; ‘0’ disables serial peripheral interface0 interrupt
- ES: ‘1’ enables serial port 0 interrupt; ‘0’ disables serial port0 interrupt
- ET1: ‘1’ enables timer 1 interrupt; ‘0’ disables timer 1 interrupt
- EX1: ‘1’ enables external interrupt 1; ‘0’ disables external interrupt 1
- ET0: ‘1’ enables timer 0 interrupt; ‘0’ disables timer 0 interrupt
- EX0: ‘1’ enables external interrupt 0; ‘0’ disables external interrupt 0

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
IE2	--	--	--	--	ESPI1	--	--	ES1

- --: Reserved bit
- ESPI1: ‘1’ enables serial peripheral interface1 interrupt; ‘0’ disables serial peripheral interface1 interrupt.
- ES1: ‘1’ enables serial port 1 interrupt; ‘0’ disables serial port1 interrupt

5.6.2 Interrupt Priorities

Each interrupt source can also be individually programmed to four priority levels by setting or clearing two bits in the SFRs named IP, IPH and IP2HL (Interrupt Priority). Figure 5-11 shows how the IE, IE2, IP, IPH and IP2HL registers and the polling sequence work to determine the interrupt to be served. The IP, IPH, IP2HL and IE2 registers contain a number of unimplemented bits, these bits are reserved and user software should not write to these positions.

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
IP	--	PT2	PSPI	PS0	PT1	PX1	PT0	PX0

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
IPH	--	PT2H	PSPIH	PS0H	PT1H	PX1H	PT0H	PX0H

- -- : Reserved bit
- [IPH(6),IP(6)] : PT2 defines timer 2 interrupt priority level
- [IPH(5),IP(5)] : PSPI defines serial peripheral interface SPI0 interrupt priority level
- [IPH(4),IP(4)] : PS0 defines serial port 0 interrupt priority level
- [IPH(3),IP(3)] : PT1 defines timer 1 interrupt priority level
- [IPH(2),IP(2)] : PX1 defines external interrupt 1 priority level
- [IPH(1),IP(1)] : PT0 defines timer 0 interrupt priority level
- [IPH(0),IP(0)] : PX0 defines external interrupt 0 priority level

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
IP2HL	PSPI1_1	PSPI1_0	PS2_1	PS2_0	--	--	PS1_1	PS1_0

- -- : Reserved bit
- IP2HL(7:6) : PSPI1 defines serial peripheral port 1 SPI1 priority level
- IP2HL(1:0) : PS1 defines serial port 1 priority level

A low-priority interrupt can only be interrupted by a higher priority interrupt (but neither by another low-priority nor an equal-priority interrupt).

If two interrupt requests of different priority levels are received simultaneously, the request of higher priority is served. If interrupt requests of the same priority level are received simultaneously, an internal polling sequence determines which request is served. Thus within each priority level there is a second priority structure determined by the polling sequence as follows:

0. IE0 (highest priority)
1. TF0
2. IE1
3. TF1
4. RI+TI
5. SPI0
6. TF2+EXF2
7. RI1+TI1
9. RI2+TI2
10. SPI1 (lowest priority)

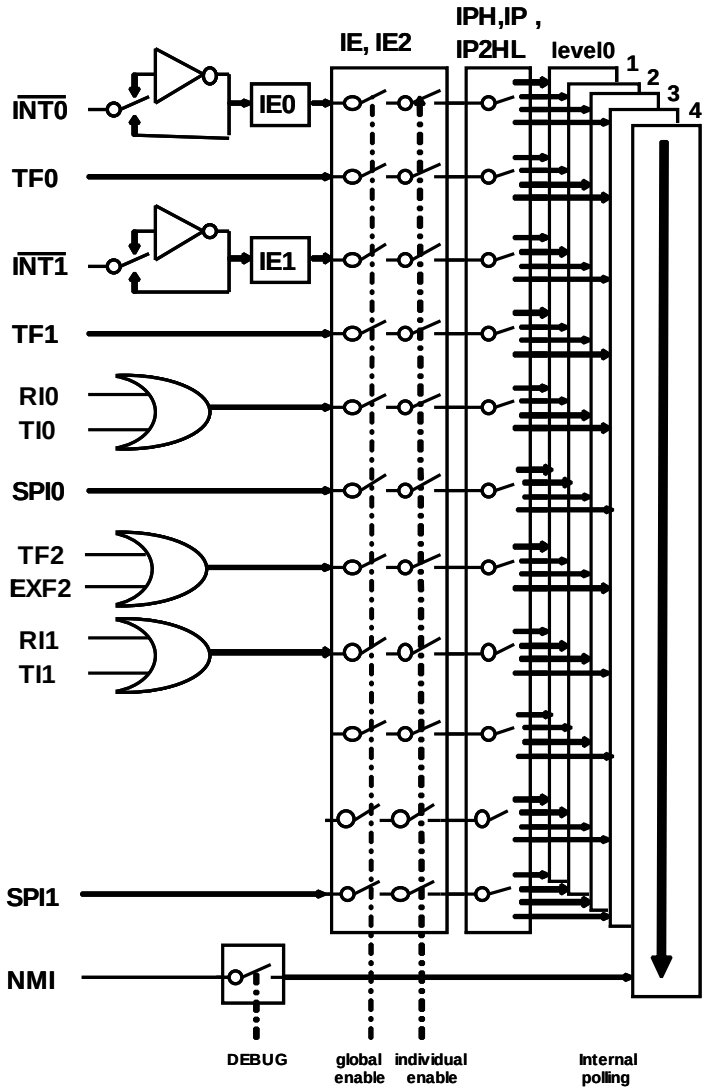
When in operation mode, all the interrupt flags are latched into the interrupt control system during state 2 of every machine cycle. The samples are polled during the following machine cycle. If the flag for an enabled interrupt is found to be set (‘1’), the interrupt system generates an LCALL to the appropriate location in Program Memory, unless some other condition blocks the interrupt. Several conditions can block an interrupt, for example when an interrupt of equal or higher priority level is already in progress.

The hardware-generated LCALL causes the value in the Program Counter to be pushed into the stack, and reloads the PC with the beginning address of the service routine. As previously noted the service routine for each interrupt begins at a fixed location.

Only the Program Counter is automatically pushed onto the stack, not the PSW or any other register.

Having only the PC automatically saved it is a programmer task to save other registers if necessary

Figure 5-11. Interrupt Priorities diagram



5.6.3 Interrupt Handling

External Interrupts INT0 and INT1 can each be either level-activated or transition-activated, depending on bits IT0 and IT1 in Register TCON. When external interrupts are transition-activated the INTx pin must show a high level in one cycle and a low level in the next cycle to generate the interrupt, so that an input high or low should hold for at least 1 machine cycle to ensure sampling. The flags that actually generate these interrupts are bits IE0 and IE1 in TCON. The flags are cleared by hardware when the service routine is vectored only if the interrupts are transition-activated. When the interrupts are level-activated the interrupt flags are controlled by the external source, so that the external source has to deactivate the request before the interrupt service routine is completed, or else another interrupt will be generated. The response time is always more than 3 machine cycles and less than 7 machine cycles depending on program execution and interrupt status (see Figure 5-12).

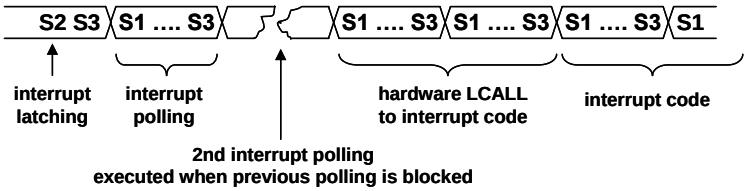
Timer 0 and Timer 1 Interrupts are generated by TF0 and TF1, which are set by a rollover. The flags are cleared by the on-chip hardware when the service routines are vectored.

Timer 2 interrupt (TF2) is user configurable, and depending on configuration flags will be cleared by hardware or by user software.

Serial Ports Interrupts are generated by the logical OR of RI and TI. They must be cleared by software.

SPI interrupts must be cleared by software.

Figure 5-12. Interrupt Handling



All of the bits that generate interrupts can be set or cleared by software. That is, interrupts can be generated or pending interrupts can be canceled by software. Each of these interrupt sources can be individually enabled or disabled by setting or clearing a bit in Special Function Register IE. IE also contains a global disable bit, EA, which disables all interrupts at once.

The interrupt flags are sampled at every machine cycle. The samples are polled during the following machine cycle. If one of the flags was in a set condition at the preceding cycle, the polling cycle will find it and the interrupt system will generate an LCALL to the appropriate service routine, provided this hardware-generated LCALL is not blocked by any of the following conditions:

- Condition 1. An interrupt of equal or higher priority level is already in progress.
- Condition 2. The current (polling) cycle is not the final cycle in the execution of the instruction in progress.
- Condition 3. The instruction in progress is RETI or any write to the IE or IP registers.

Any of these three conditions will block the generation of the LCALL to the interrupt service routine. Condition 2 ensures that the instruction in progress will be completed before vectoring to any service routine. Condition 3 ensures that if the instruction in progress is RETI or any access to IE or IP, then at least one more instruction will be executed before any interrupt is vectored to.

The polling cycle is repeated with each machine cycle, and the values polled are the values that were present at the previous machine cycle. The polling cycle/LCALL sequence is illustrated in Figure 5-12.

The hardware-generated LCALL executes a program jump to fixed program entries as shown below:

Table 5-12. Fixed Program entries

Source	Vector Address
IE0	0003H
TF0	000BH
IE1	0013H
TF1	001BH
RI0+TI0	0023H
SPI0	002BH
TF2	0033H
RI1+TI1	003BH
SPI1	0053H

Execution proceeds from that location until the RETI instruction is encountered then the interrupted program continues from where it left off.

5.7 I/O Ports

The ports P1, P3, P4 and P5 in the ADD8051C3A are bidirectional. Each port consists of a register (Special Function Registers P1, P3, P4, P5), an output driver, and an input buffer.

Pins in ports 3, 4 and 5 are multifunctional. They are port pins and also serve as input or output for the microcontroller peripherals:

Table 5-13. P3, P4, P5 ports Alternate Functions

Pin	Alternate Function
P3.0	RxD (serial port 0 input)
P3.1	TxD (serial port 0 output)
P3.2	INT0 (external interrupt 0)
P3.3	INT1 (external interrupt 1)
P3.4	T0 (timer 0 external input)
P3.5	T1 (timer 1 external input)
P3.6	WR(external data memory write strobe)
P4.2	SS1 (SPI1 slave select input)
P4.3	SPICLK1 (SPI1 clock input/output)
P4.4	MOSI1 (SPI1 master out / slave in data)
P4.5	MISO1 (SPI1 master in / slave out data
P4.6	T2 (timer 2 input/output)
P5.0	SS (SPI0 slave select input)
P5.1	SPICLK (SPI0 clock input/output)
P5.2	MOSI (SPI0 master output / slave input data)
P5.3	MISO (SPI0 master input / slave output data)
P5.4	RxD (serial port 1 input)

P5.5	TxD (serial port 1 output)

Note: To use an Alternate Function the corresponding bit (or bits) in the SFR must contain a 1.

5.7.2 I/O Configurations

Figure 5-13 and Figure 5-14 show a functional diagram of bit register and I/O buffer in each of the four ports. The level of the port pin is placed on the internal bus and instructions can read the port pin value or the SFR register value.

If a bit in a register with AOF (Alternate Output Function) contains a 1, then the output level is controlled by the signal labeled “Alternate Output function”.

Figure 5-13. Pins with AOF structure

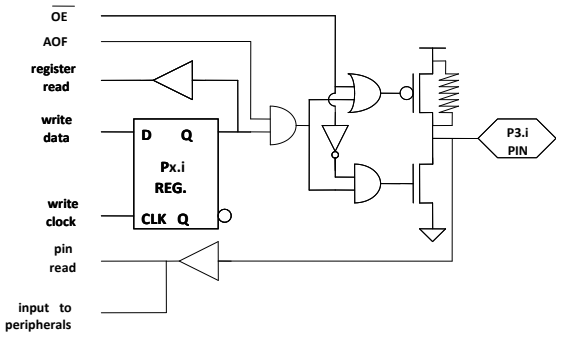
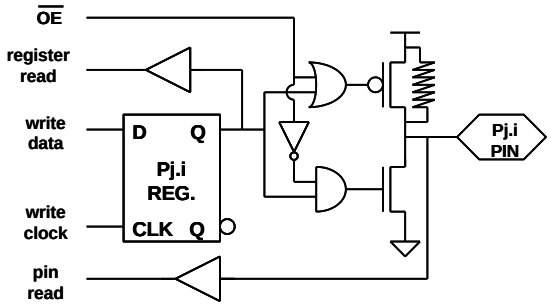


Figure 5-14. Pins without AOF structure



Pseudo-bidirectional mode

Ports 1, 3, 4 and 5 have internal pull-ups. Each I/O line can be independently used as an input or an output. To be used as an input, the port bit register must contain a ‘1’; When an SFR port bit is set to ‘1’ the PMOS port driver is turned on one clock cycle and then is turned off. Then, the pin is pulled high by a weak internal pull-up, and can be pulled low by an external source. In the ADD8051C3A included in ATPL210A, the pull-up consists of 33Kohm resistors.

P1, P3, P4 and P5 registers are set to FFH by default (configured as inputs).

Push-Pull mode

Ports 3, 4 and 5 can be configured as Push-Pull output ports using P3M, P4M and P5M SFRs respectively.

Setting a bit in these SFRs to `1’ configures the corresponding port as Push-Pull mode.

Setting a bit in these SFRs to ‘0’ configures the corresponding port as Pseudo-bidirectional.

P3M, P4M and P5M are set to 00H by default (configured in Pseudo-bidirectional mode).

5.7.3 Read-Modify-Write Feature

Some instructions that read a port read the register and others read the pin. Only the read-modify-write instruction read the register. The instructions that read the register rather than the pin are the ones that read a value, possibly change it, and then rewrite it to the register. When the destination operand is a port, or a port bit, these instructions read the register rather than the pin: ANL, ORL, XRL, JBC, CPL, INC, DEC, DJNZ, MOV PX.Y,C, CLR PX.Y, SET PX.Y. The last three instructions are bit modify instructions. They read the port byte, all 8 bits, modify the target bit, then the modified byte is written to the register.

The-modify-write instructions are directed to the register to avoid a possible misinterpretation of the voltage level at the pin. When a port register is set to ‘1’ to using the pin as input an external source can be pulling the pin to ‘0’, if we need to know the configuration of the port we need to read the register and to know the state of the external source we need to read the pin.

5.7.4 Accessing External Memories

Accesses to external memory are of two types: accesses to external program code and accesses to external program data. Both share ports ADDRESS and DATA buses. Accesses to external program code use signal PSEN (program store enable) as the read strobe. Accesses to external program data use RD(P3.7) or WR(P3.6) to strobe the memory.

Fetches from external program memory always use a 16-bit logical address, fetches to external data can use 16-bit or 8-bit logical address.

5.8 Debug Mode

Debug mode is active when DBG pin is set to ‘1’. Debug mode is intended to implement software debugging tools.

DEBUG digital input is internally connected to DBG signal of the ADD8051C3A microcontroller

When in debug mode, a program can modify its own code. Thus, user **should not** work in debug mode unless explicitly stated otherwise.

6. Timers

The ADD8051C3A has a full set of timers, listed below:

- Two 16-bit configurable Timers/Counters: Timer 0 and Timer 1 that can be configured as timers or event counters (Figure 29). In ATPL210A this timers are internal devices, thus external signals counter configuration is not available. Nevertheless, the complete device operation is described below in order to know the suitable configuration registers values
- Two 8-bit auto reload counters Timers 11 and 12, used as baud rate generators for uarts 0 and 1 respectively (see Standard Serial Interfaces section).
- One 8-bit, with 18-bit preescaler, watchdog timer: Timer 3 (see Watchdog (timer 3) section).
- One 16-bit Timer/Counter Timer 2 with capture reload hardware (see 6.2 section).

6.1 Timer 0 and Timer 1

Timers 0 and 1 have 2-bit preescalers. These preescalers can be programmed to slow down count rate. The reset value set the preescaler values to the slowest count rate and the real count rate of the timers is the same than the timers in an “8051 legacy” microcontroller (12 clock cycles per machine cycle).

TPR, TMOD and TCON registers described below are involved in timer 0 and timer 1 management.

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
TPR	--	--	T1P1	T1P0	T01P1	T01P0	T0P1	T0P0

- --: Reserved bit
- T1P(1:0): Rollover value for T1 preescaler when T1 is in modes 0, 1, 2 and 3
- T01P(1:0): Rollover value for TH0 preescaler when T0 is in mode 3
- T0P(1:0): Rollover value for T0 preescaler when T0 is in modes 0, 1, and 2, and for TL0 preescaler when T0 is in mode 3

When T0 or T1 are in the timer function, the registers are incremented every 1, 2, 3 or 4 machine cycles. Since a machine cycle consists of 3 oscillator periods, the count rate is 1/3, 1/6, 1/9 or 1/12 of the oscillator frequency.

In the counter function, the register is incremented in response to a 1-to-0 transition at its corresponding external input signal, T0(P3.4) or T1(P3.5). When in counter function, the external input is sampled once every machine cycle. When the samples show a high in one cycle and a low in the next cycle (1-to-0 transition), the count is incremented. The register value is updated during the cycle following the one in which the transition was detected, the maximum count rate is 1/2 of the machine cycle frequency.

Either the timer or counter function is selected by control bit C/T in the Special Function Register TMOD where T0 and T1 have four operating modes, which are selected by bit-pairs (M1, M0) in TMOD.

	TIMER 1				TIMER 0			
Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
TMOD	GATE	C/T	M1	M0	GATE	C/T	M1	M0

- GATE:
  - ‘1’: Enables gate control. Device count is enabled only when both ‘INTi’ pin and “TCON.TRi” are high.
  - ‘0’: Disables gate control. Device count is enabled when “TCON.TRi” is high.

- C/T:
  - ‘1’: Counter operation mode. Input from Ti input pin
  - ‘0’: Timer operation mode. Input from internal clock system
- M(1:0): Operation modes
  - “00”: Mode 0: 8048 13bit timer, TLi used as 5 bit preescaler
  - “01”: Mode 1: 16 bit timer/counter composed by concatenated THi and TLi
  - “10”: Mode 2: 8 bit with auto reload (TLi <= THi) timer/counter
  - “11”: Mode 3: timer 1 stopped
    - Timer 0 runs as two 8-bit independent timer/counters:
      - TL0 is a timer/counter controlled by timer 0 control bits
      - TH0 is a timer controlled by timer 1 control bits

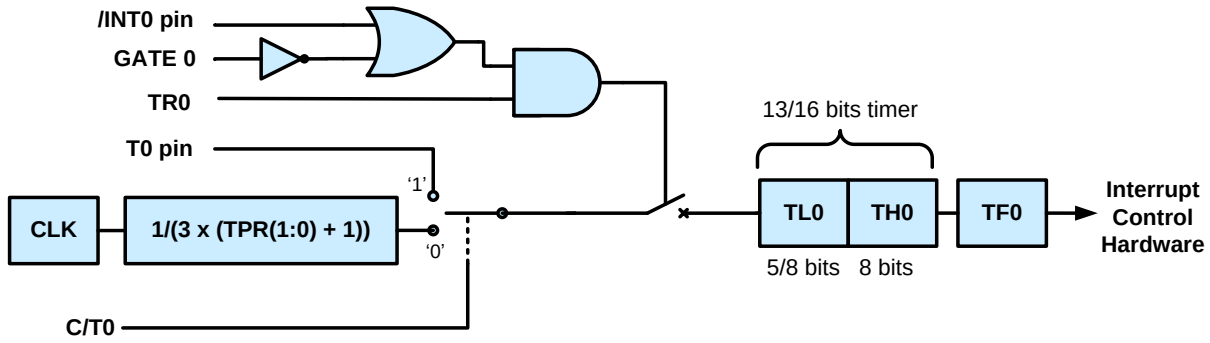
Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
TCON	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

- TF1: Timer 1 overflow flag, set by hardware overflow and cleared by software or by hardware when interrupt routine is vectored
- TR1: Timer 1 run control bit, when set turns on timer/counter 1
- TF0: Timer 0 overflow flag, set by hardware overflow and cleared by software or by hardware when interrupt routine is vectored
- TR0: Timer 0 run control bit, when set turns on timer/counter 0
- IE1: Interrupt 1 edge flag, set by hardware when an external interrupt edge is detected and cleared by software or by hardware when interrupt routine is vectored
- IT1: Interrupt 1 type control bit. “’1’/’0’” to specify “falling edge/low level” trigger for external interruption 1
- IE0: Interrupt 0 edge flag, set by hardware when an external interrupt edge is detected and cleared by software or by hardware when interrupt routine is vectored
- IT0: Interrupt 0 type control bit. “’1’/’0’” to specify “falling edge/low level” trigger for external interruption 0

6.1.1 Timer Mode 0

The timer (0 or 1) into Mode 0 looks like a 13-bit timer: an 8-bit counter (all 8 bits of THx) with a divide-by-32 preescaler (the lower 5 bits of TLx), see Figure 6-1. As the count rolls over from all 1s to all 0s, it sets the Timer interrupt flag TFi. The counter input is enabled to the Timer when TRi=1 and either GATE=0 or INT1=1. When GATE=1 the Timer can be controlled by external input INTx, to facilitate pulse width measurements. TRx is a control bit in the Special Function Register TCON.

Figure 6-1. Timer 0 (or Timer 1) in modes 0,1



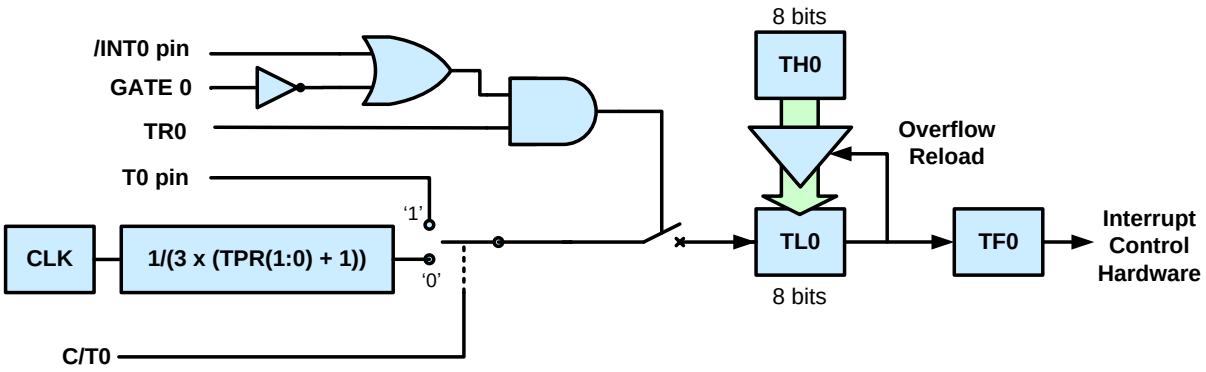
6.1.2 Timer Mode 1

In Mode 1 the timer (0 or 1) runs in the same way than Mode 0 but they are 16 bits-width and TLx is not used as a preescaler.

6.1.3 Timer Mode 2

Mode 2 configures the timer (0 or 1) register as an 8-bit Counter (TLx) with automatic reload of THx value. THx value is preset by software.

Figure 6-2. Timer 0 (or Timer 1) in mode 2



6.1.4 Timer Mode 3

In Mode 3 Timer 0 registers TL0 and TH0 are configured as two separate 8-bit counters, TL0 and TH0.

TL0 uses the Timer 0 control bits and can be configured either in timer or in counter mode.

TH0 is locked into a timer function (counting machine cycles) and is controlled by Timer 1 control bits TR1 and TF1.

TH0 controls the Timer 1 interrupt.

Timer 1 in Mode 3 holds its count. With Timer 0 in Mode 3, the number of timer/counters in the 80C51 is increased to three. When Timer 0 is in Mode 3, Timer 1 can be used by the serial port as a baud rate generator, or in any application not requiring an interrupt.

6.2 Timer 2

Timer 2 is a 16-bit timer/counter. The count is maintained by two eight-bit timer registers, TH2 and TL2, connected in cascade. T2MOD and T2CON registers described below control the operation of timer 2.

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
T2MOD	--	--	TF2SC	EXF2SC	T2PR1	T2PR0	T2OE	DCEN

- --: Reserved bit
- TF2SC: Timer 2 overflow flag software control. When set to ‘1’, TF2 flag must be cleared by software. When cleared to ‘0’, TF2 flag is cleared by hardware when interrupt routine is vectored
- EXF2SC: External Interrupt 2 flag software control. When set to ‘1’, EXF2 flag must be cleared by software. When cleared to ‘0’, EXF2 flag is cleared by hardware when interrupt routine is vectored
- T2PR(1:0): Timer 2 preescaler
- T2OE: Timer 2 output enable bit. In the timer 2 clock-out mode connects the programmable clock input to the external pin T2
- DCEN: Down count enable bit, configures timer 2 as an up/down counter

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
T2CON	TF2	EXF2	SSI1	SSI0	EXEN2	TR2	C/T2	CP/RL2

- TF2: Timer 2 overflow flag, Set by timer 2 overflow. It must be cleared by software. TF2 is not set when timer 2 is in baud rate generator mode
- EXF2: Timer 2 external flag. If EXEN2=’1’ capture or reload caused by a negative transition on T2EX sets EXF2. EXF2 does not cause an interrupt in up/down counter mode (DCEN=’1’)
- SSI(1:0): Select T2 as baud rate generator for serial port
  - “00”: none
  - “01”: USART0
  - “10”: USART1
  - “11”: USART2
- EXEN2: Timer 2 external enable bit. Setting EXEN2 causes a capture or reload to occur as a result of a negative transition on T2EX unless timer 2 is being used as baud rate generator. Clearing EXEN2 causes timer 2 to ignore events at T2EX
- TR2: Timer 2 run control bit. Setting this bit starts the timer
- C/T2: Timer 2 counter/timer select. When set to ‘1’, it selects counter operation then timer 2 counts negative transitions on system clock
- CP/RL2: When set to ‘1’, captures occur on negative transitions at T2EX if EXEN=’1’; When cleared to ‘0’, auto reload occurs on timer 2 overflow or negative transitions at T2EX if EXEN2=`1`. If RCLK=’1’ or TCLK=’1’ the CP/RL2 bit is ignored

Timer 2 provides the following operating modes: capture mode, auto-reload mode, baud rate generator mode, and programmable clock-out mode. Select the operating mode with T2MOD and T2CON register bits as shown in Table 6-1. Auto-reload is the default mode. Setting T2CON(5:4) selects the baud rate generator mode.

Table 6-1. Timer 2 operating modes

Mode	T2CON(5) OR T2CON(4)	CP/RL2	T2OE
Auto-reload	0	0	0
Programmable clock-out	0	0	1
Capture	0	1	0
stopped	0	1	1
Baud Rate generator	1	X	X

Timer 2 operation is similar to timer 0 and timer 1. C/T2 selects between timer operation mode (internal clock input) or counter operation mode (external pin T2 as the time register input). Setting TR2 allows TL2 to be incremented by the selected input.

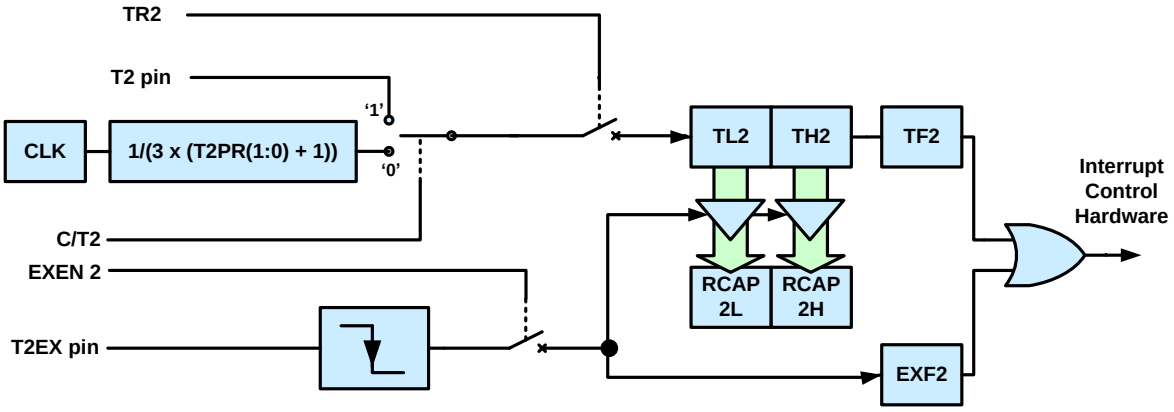
The timer 2 related interrupt flags (TF2, EXF2) can be configured to be cleared by hardware when the interrupt routine is vectored to, or by user software using TF2SC and EXF2SC.

The operating modes are described in the following paragraphs.

6.2.2 Capture mode

In the capture mode, timer 2 operates as a 16-bit timer or counter. An overflow condition sets bit TF2, which can be used to request an interrupt. Setting the external enable bit EXEN2 allows the RCAP2H and RCAP2L registers to capture the current value in timer registers TH2 and TL2 in response to a 1-to-0 transition at external input T2EX. The transition at T2EX also sets bit EXF2 in T2CON. The EXF2 bit, like TF2, can generate an interrupt.

Figure 6-3. Timer 2 capture mode



6.2.3 Auto-Reload mode

The auto-reload mode configures timer 2 as a 16-bit timer or event counter with automatic reload. The timer operates as an up counter or as an up/down counter, as determined by the down counter enable bit (DCEN). When reset occurs, DCEN is cleared, so in the auto-reload mode, timer 2 in configured as an up counter by default.

- **Up Counter Operation**

  When DCEN = 0, timer 2 operates as an up counter.

The external enable bit EXEN2 in the T2CON register provides two options. If EXEN2 = 0, timer 2 counts up to FFFFH and sets the TF2 overflow flag. The overflow condition loads the 16-bit value in the reload/capture registers (RCAP2H, RCAP2L) into the timer registers (TH2, TL2). The values in RCAP2H and RCAP2L are preset by software.

If EXEN2 = 1, the timer registers are reloaded by either a timer overflow or a high-to-low transition at external input T2EX. This transition also sets the EXF2 bit in the T2CON register.

Either TF2 or EXF2 bit can generate a timer 2 interrupt request.

- **Up/Down Counter Operation**

When DCEN = 1, timer 2 operates as an up/down counter.

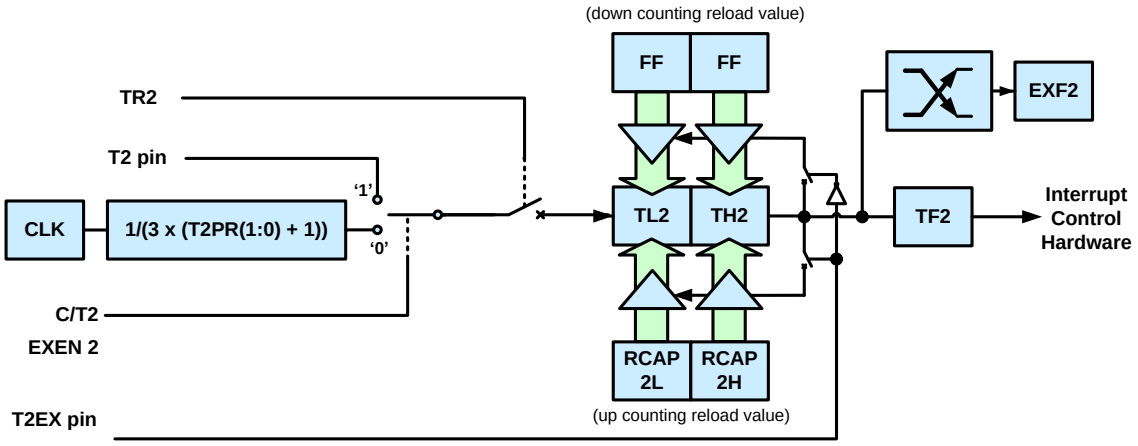
External pin T2EX controls the direction of the count.

When T2EX is high, timer 2 counts up. The timer overflow occurs at FFFFH which sets the timer 2 overflow flag (TF2) and generates an interrupt request. The overflow also causes the 16-bit value in RCAP2H and RCAP2L to be loaded into the timer registers TH2 and TL2.

When T2EX is low, timer 2 counts down. Timer underflow occurs when the count in the timer registers (TH2, TL2) equals the value stored in RCAP2H and RCAP2L. The underflow sets the TF2 bit and reloads FFFFH into the timer registers.

The EXF2 bit toggles when timer 2 overflows or underflows changing the direction of the count. When timer 2 operates as an up/down counter, EXF2 does not generate an interrupt. This bit can be used to provide 17-bit resolution.

Table 6-2. Timer 2 Auto-reload mode



6.2.4 Clock-Out mode

In the clock-out mode, timer 2 operates as a 50%-duty-cycle, variable-frequency clock. The input clock increments TL2 at frequency FOSC/2 (where FOSC=20MHz). The timer repeatedly counts to overflow from a preloaded value. At overflow, the contents of the RCAP2H and RCAP2L registers are loaded into TH2/TL2. In this mode, timer 2 overflows do not generate interrupts. The formula gives the clock-out frequency as a function of the system oscillator frequency and the value in the RCAP2H and RCAP2L registers:

For a 20 MHz system clock, timer 2 has a programmable frequency range of 12.7 Hz to 3.3 MHz. The generated clock signal is brought out to the T2 pin (P4.6).

Timer 2 is programmed for the clock-out mode as follows:

1. Set the T2OE bit in T2MOD. This gates the timer register overflow to the ÷2 counter.

2. Clear the C/T2 bit in T2CON to select FOSC/3*(T2PRE(1:0)+1) as the timer input signal. This also gates the output of the ÷2 counter to pin T2.

3. Determine the 16-bit reload value from the formula and enter in the RCAP2H/RCAP2L registers.

4. Enter a 16-bit initial value in timer register TH2/TL2. This can be the same as the reload value, or different, depending on the application.

5. To start the timer, set the TR2 run control bit in T2CON.

Clock-out mode timer 2 operation is similar to timer 2 operating as a baud rate generator. Moreover, it is possible to use timer 2 as a baud rate generator and a clock generator simultaneously. For this configuration, the baud rates and clock frequencies are not independent since both functions use the values in the RCAP2H and RCAP2L registers

The baud rate is expressed by the following formula:

6.2.5 Baud rate Generator mode

This mode configures timer 2 as a baud rate generator for its use with the serial ports. Select this mode by setting the SSI(1:0) bits in T2CON.

Timer 2 may be selected as the baud rate generator for the transmitter and/or receiver in modes 1 and 3. The timer 2 baud rate generator mode is similar to the auto-reload mode. A rollover in the TH2 register reloads registers TH2 and TL2 with the 16-bit value in registers RCAP2H and RCAP2L, which are preset by software.

The timer 2 baud rate is expressed by the following formula:

Where “BAUD2” is the bit in the Special Function Register CONF and RCAP2(H,L) denotes the content of RCAP2H and RCAP2L taken as a 16-bit unsigned integer.

To select timer 2 as baud rate generator for transmission and reception, program the SSI(1:0) bits in the T2CON register as shown in Table 6-1. Setting SSI(1) and/or SSI(0) puts timer 2 into its baud rate generator mode. In this mode, a rollover in the TH2 register does not set the TF2 bit in the T2CON register. Also, a high-to-low transition at the T2EX pin sets the EXF2 bit in the T2CON register but does not cause a reload from (RCAP2H, RCAP2L) to (TH2, TL2).

User shall use the T2EX pin as an additional external interrupt by setting the EXEN2 bit in T2CON.

Turn the timer off (clear the TR2 bit in the T2CON register) before accessing registers TH2, TL2, RCAP2H, and RCAP2L

User shall configure timer 2 as a timer or a counter. In most applications, it is configured for timer operation (i.e., the C/T2 bit is clear in the T2CON register).

Note that timer 2 increments every 1, 2, 3 or 4 state times (3TOSC) when it is in the baud rate generator mode.

When timer 2 is configured as a timer and in baud rate generator mode, do not read or write the TH2 or TL2 registers. The timer is being incremented every state time, and the results of a read or write may not be accurate. In addition, user shall read, but not write to, the RCAP2 registers; a write may overlap a reload and cause write and/or reload errors.

Table 6-3 lists commonly used baud rates and shows how they are generated by timer 2.

Table 6-3. Timer 2 generated baud rates with a 20MHz oscillator

Baud rate	T2PRE	RCAP2H	RCAP2L
230400	0	FF	FE
115200	0	FF	FC
57600	0	FF	F9
38400	0	FF	F5
19200	0	FF	EA
9600	0	FF	D5
4800	0	FF	A9
2400	0	FF	52
1200	0	FE	A5
600	0	FD	4A
300	0	FA	93
150	0	F5	26
110	0	F1	34

6.3 Watchdog (timer 3)

The watchdog timer includes a 18-bit preescaler which is set to 0 when T3 is reloaded.

T3 is active only when /EWDG pin is tied to '0'.

T3 must be reloaded by software to avoid T3 rollover and program restart.

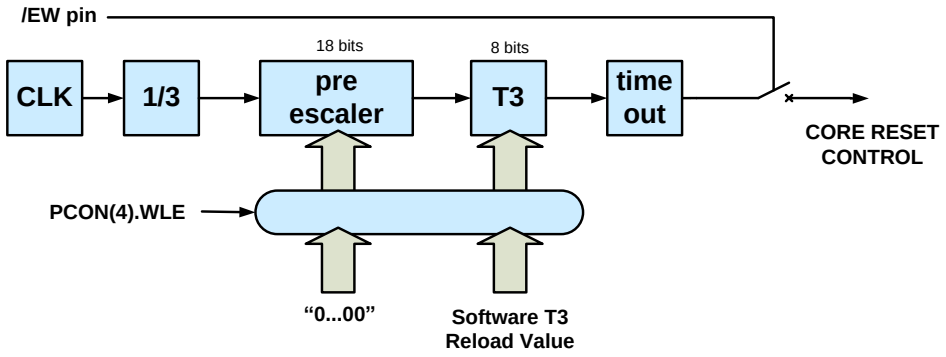
Reload of T3 is allowed only if WLE watchdog load enable (PCON(4)) is set.

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
PCON	SMOD	--	--	WLE	GF1	GF0	PD	IDL

- SMOD: Doubles baud rate bit when timer 1 is used as time generator and serial port is in modes 1,2 or 3
- --: Reserved bit
- WLE: Watchdog load enable. It must be set by software to enable T3 reload. WLE is reset by hardware 13 machine cycles after been set by user software.
- GF1: General purpose flag bit, user programmable
- GF0: General purpose flag bit, user programmable
- PD: Sets power down mode
- IDL: Sets idle mode

Reload of T3 register automatically resets the preescaler and WLE

Figure 6-4. Watchdog timer



7. Standard Serial Interfaces

Two full duplex (simultaneous transmission/reception) serial ports are implemented in ATPL210. The serial port reception and transmission registers are both accessed at Special Function Register SBUF(0,1). Writing to SBUF loads the transmit register and starts transmission, while reading SBUF accesses a physically separate receive register. The reception register is buffered and it can start reception of a second byte before a previously received byte has been read from the register. When a second byte is fully received the first one is lost.

The serial port control and status register is the Special Function Register SCON(0,1).

This register contains the mode selection bits, the 9th data bit for transmit and receive (TB8 and RB8), and the serial port interrupt bits (TI and RI).

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
SCON	SM0	SM1	SM2	REN	TB8	RB8	TI	RI

- SM(0:1):
  - “00”: Mode 0: Shift register mode; baud rate=1/12 fclk
  - “01”: Mode 1: 8-bit UART mode; baud rate= T1 overflow
  - “10”: Mode 2: 9-bit UART mode; baud rate= 1/32 or 1/64 fclk
  - “11”: Mode 3: 9-bit UART mode; baud rate= T1 overflow
- SM2: Reception control. In modes 2 and 3 with SM2=’1’, RI will be ‘0’ if the 9th received bit is ‘0’; In mode 1 with SM2=’1’, RI will be ‘0’ if the received stop bit is not correct; In mode 0 SM2 keeps a ‘0’.
- REN: Serial port reception enable
- TB8: 9th transmitted bit in modes 2 and 3
- RB8: 9th received bit in modes 2 and 3, received stop bit in mode 1 when SM2=’0’
- TI: Transmission interrupt flag. Set by hardware at the end of transmission, must be cleared by software
- RI: Reception interrupt flag. Set by hardware at the end of transmission, must be cleared by software

The serial port can operate in 4 modes. Transmission is always initiated by any instruction that writes SBUF, and TI flag is set after transmission. Reception is initiated in Mode 0 by the condition RI=0 and REN=1, and in the other modes by the incoming start bit if REN=1. In Modes 2 and 3, 9 data bits are received. The 9th one goes into RB8 and then comes a stop bit.

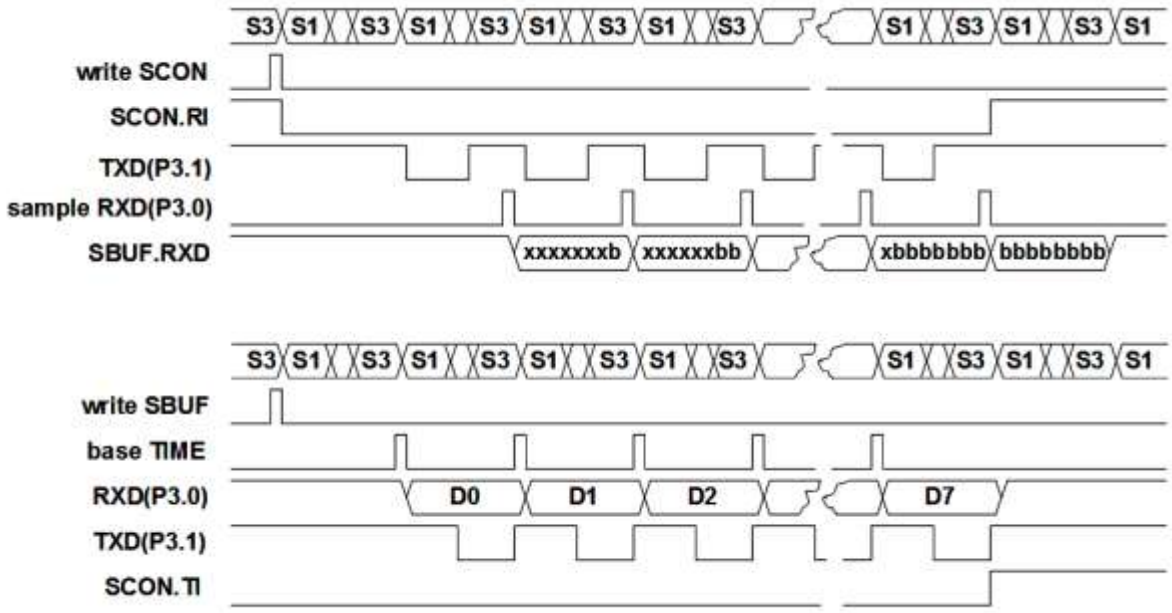
The port can be programmed such that when the stop bit is received, the serial port interrupt will be activated only if RB8 = 1. This feature is enabled by setting bit SM2 in SCON.

7.1 Serial Port modes

7.1.1 Mode 0 (shift register mode)

Serial data inputs and exits through RxD. TxD outputs the shift clock. Eight bits are transmitted/received (LSB first). The baud rate is fixed at 1/12 the oscillator frequency. The internal timing is such that four full machine cycles will elapse between “write to SBUF” command and activating the transmission; transmission ends in the 36th machine cycle after “write to SBUF”. Reception is initiated by the conditions REN = 1 and RI = 0, and after 36 machine cycles reception finish and RI is set to ‘1’.

Figure 7-1. Serial Port reception/emission mode 0



7.1.2 Mode 1 (8-bit UART)

In this mode, 10 bits are transmitted (through TxD) or received (through RxD): a start bit (0), 8 data bits (LSB first), and a stop bit (1). In reception, the stop bit goes into RB8 in SFR SCON. The baud rate is controlled by T1 rollover. Transmission is initiated by any instruction that uses SBUF as a destination register and the bit times are synchronized to T1 rollover. Reception is initiated by a detected 1-to-0 transition at RxD, RxD is sampled at a rate of 16 times each bit time and the value accepted is the value that was seen in at least 2 of the 3 central samples (7th, 8th, and 9th).

A correct reception loads SBUF and RB8 and sets RI, this is achieved when at the end of the receive process the condition (RI='0' and (SM2='0' or received_stop_bit='1')) is met. Otherwise, if the condition is not met, the received frame is irretrievably lost.

7.1.3 Modes 2 and 3 (9-bit UART)

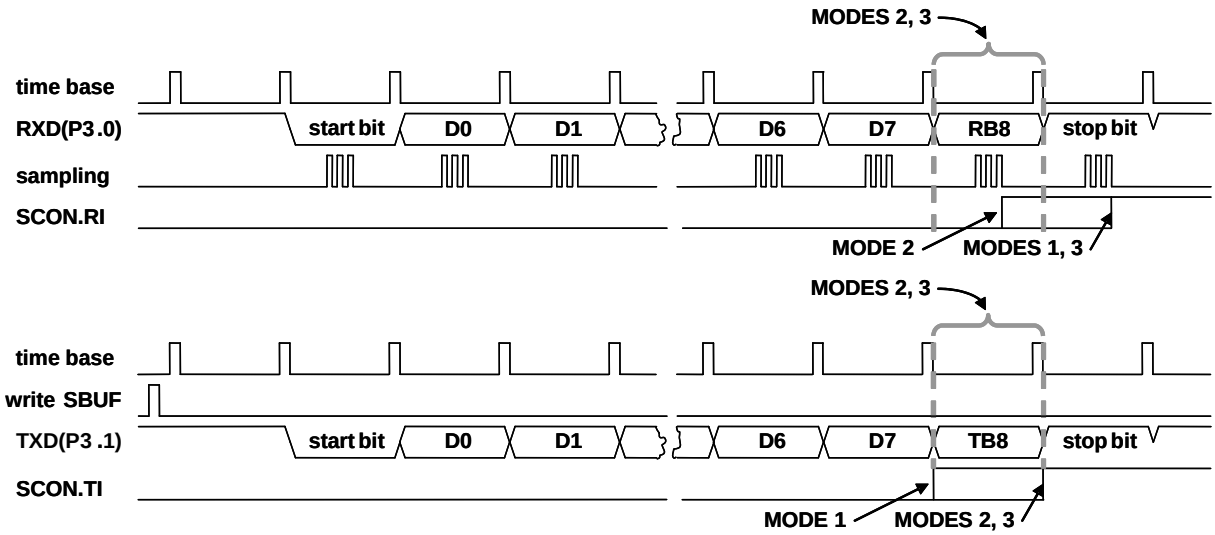
In this mode, 11 bits are transmitted (through TxD) or received (through RxD): start bit (0), 8 data bits (LSB first), a programmable 9th data bit (TB8), and a stop bit (1). The 9th transmitted data bit is loaded with TB8 value from SFR SCON. On reception, the 9th data bit goes into RB8 in SFR SCON. The stop bit is ignored. The baud rate is programmable to 1/32 or 1/64 the oscillator frequency. Mode 3 is the same as Mode 2 with programmable baud-rate controlled by T1 rollover.

The transmission begins with a "write to SUBF" instruction and finish at the 11th rollover after "write to SUBF", TI is set when transmission ends.

Reception is initiated by detecting a high to low level transition at RxD. For this purpose RxD is sampled at a rate of 16 times each bit. The value accepted is the value that was seen in at least 2 out of 3 central samples. When reception ends SBUF and RB8 are loaded and RI is set. A frame is correctly received if, and only if, the following condition is met at the time the last bit is received (RI=0 and (SM2='0' or 9th_data_bit=1).

Otherwise, if the condition is not met the received frame is irretrievably lost and RI is not set.

Figure 7-2. Serial Port reception/emission modes 1,2,3.



7.2 Serial Port Timers (Timers 11 &12)

Timers 11 and 12 are similar to T1 configured as 8-bit timer in auto reload mode (mode 2). These timers are used as baud rate generators for uarts 0 and 1 respectively.

These timers are **not connected to the interruption system hardware**. They have not preescaler counter and their values are incremented every machine cycle (3 clock cycles).

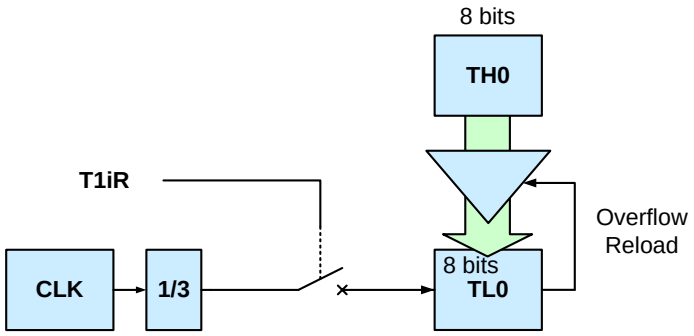
They can be used only as baud rate generators for the standard serial interfaces. When the run control bit in T1NC register (T1iR) is set the corresponding timer (T1i) starts counting machine cycles and the related SSI is controlled by the overflow of T1i.

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
T1NC	WT12	WT11	WT10	--	--	--	T12R	T11R

- WT1(2:0): read/write access control to T1 timer registers
  - “000”: TH1, TL1
  - “001”: TH11, TL11
  - “010”: TH12, TL12
  - “100”: TH14, TL14
- --: Reserved bit.
- T12R: TT2 run control, when set T12 runs and T12 overflow controls UART1 variable baud rates.
- T11R: TT1 run control, when set T11 runs and T11 overflow controls UART0 variable baud rates.

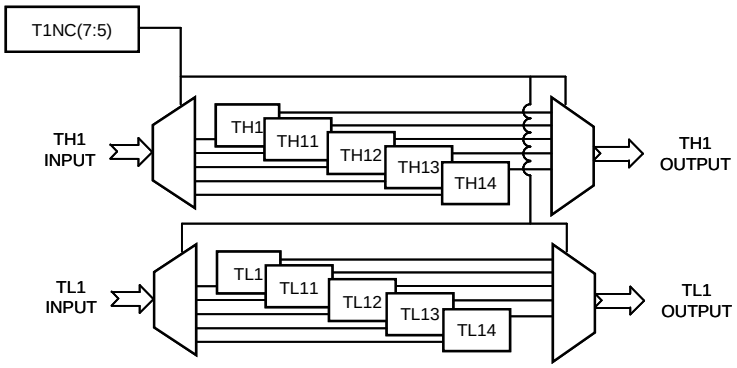
Registers TH11, TH12 and TH14 are mapped in the same address than TH1, to select the specific register to be accessed (read or write) the bits WT1(2:0) in T1NC register must be set to the desired value. TL11, TL12, TL14 and TL1 are accessed in the same way.

Figure 7-3. T1i timer



Timers T1i count machine cycles (3 clock cycles) in normal operation mode. They can be forced to count clock cycles. To enter this mode the corresponding bits in T1NP register must be set to ‘1’ (T1NP(i-1) controls T1i). Unused bits in T1NP, bits 4 to 7, are not defined. Writing to these bits has no effect, reading these bits will always get “0000”.

Figure 7-4. TH1 and TL1 registers



Mode 0

The baud rate in Mode 0 is fixed to Oscillator Frequency / 12.

Modes 1 and 3

The baud rates in Modes 1 and 3 are determined by the Timer 1 overflow rate and the value of SMOD and BAUD2 as follows:

Mode 2

The baud rate in Mode 2 depends on the value of bit SMOD in Special Function Register PCON and BAUD2 in Special Function Register CONF.

If [BAUD2,SMOD] = “00” (which is the value on reset), the baud rate is 1/64 the oscillator frequency.

If [BAUD2,SMOD] = “01” or “10”, the baud rate is 1/32 the oscillator frequency.

If [BAUD2,SMOD] = “11”, the baud rate is 1/16 the oscillator frequency

---

In the most typical applications, it is configured for “timer” operation in the auto-reload mode. In that case the baud rate is given by the formula:

When timers T11, T12, T13 or T14 are selected as timing sources the baud rate is given by the formula:

Timers T1i count machine cycles (3 clock cycles) in normal operation mode. They can be forced to count clock cycles. To enter this mode the corresponding bits in T1NP register must be set to ‘1’. When T1i are configured in this mode the baud rate is given by the formula:

In this mode 0xFF and 0xFE are not allowed as reload values. When these values are used the behavior of USART is unpredictable.

Timer 2 overflow can also be used as baud rate generator for serial ports in modes 1 and 3. See timer 2 section for more information.

Selection of T1i as timing source overrides T2 and T1 sources. Selection of T2 as timing source overrides T1 source.

8. Serial Peripheral Interfaces SPI0 and SPI1

Two Serial Peripheral Interfaces SPI0 and SPI1 are implemented in ATPL210A.

Both SPI0 and SPI1 are full-duplex, synchronous communication bus with two operation modes: master mode and slave mode. When acting as an SPI master, the microcontroller supports baud rates of up to ¼ of the clock frequency.

The main difference between SPI0 and SPI1 is that while SPI0 is a byte oriented SPI, SPI1 is a byte-buffered SPI. SPI1 byte-buffering will be explained as a special SPI capability type in section 8.3. Ignoring this buffer capability, both SPI0 and SPI1 are identical internally.

8.1 SPI description

During a byte transfer over SPI, the master and the slave simultaneously transmit (shift out) and receive (shift in) data. Master is the responsible in SPI system for transfer initialization, proper shifting and sampling of data. Every activity in the SPI environment is synchronized with the clock signal which is under master's control. A slave select line allows the master to select the desired slaves to exchange data with; not selected slave devices cannot interfere ongoing bus activities. In systems with multiple masters, slave select line on the master device can be used to detect multiple master bus contention.

SPI0 is configured and controlled using three special function registers: SPCTL, SPSTAT and SPDAT. (Similarly, SPI1 is configured and controlled by means of SPCTL1, SPSTAT1 and SPDAT1N registers)

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
SPCTL	SSIG	SPEN	DORD	MSTR	CPOL	CPHA	PSC1	PSC0

- SSIG: /SS0 slave select ignore, if set to '1' MSTR decides whether the device is a master or a slave and /SS0 pin can be used as port pin; if set to '0' and MSTR is set to '1' then the /SS pin decides whether the device is a master or a slave
- SPEN: SPI0 enabled '1' or disabled '0', when the SPI0 is disabled the SPI0 pins can be used as general I/O pins
- DORD: SPI0 data order

  '1' - the lsb of the data byte is transmitted first

  '0' - the msb of the data byte is transmitted first
- MSTR: master '1' / slave '0' mode select. See SSIG field.
- CPOL: SPI0 clock polarity

  '1' – SPICLK0 is high when idle, the leading edge of SPICLK0 is the falling edge and the trailing edge is the rising edge

  '0' – SPICLK0 is low when idle, the leading edge of SPICLK0 is the rising edge and the trailing edge is the falling edge
- CPHA: SPI0 clock phase select

  '1' - data is driven on the leading edge of SPICLK0 and is sampled on the trailing edge

  '0' - the first data bit is on the MOSI0/MISO0 line before the first SPICLK0 leading edge, data is sampled on the leading edge of SPICLK0 and driven on the trailing edge of SPICLK
- PSC(1:0): SPI0 clock rate select, determines master clock output. 8 different baud rates can be selected using PSC2[SPSTAT(5)] & PSC(1:0)[SPCTL(1:0)].

  When working as slave, PSC(2:0) value is not taken into account, nevertheless this value must be different from "000"

PSC(2:0)	clock rate
“000”	fosc/4
“001”	fosc/16
“010”	fosc/64
“011”	fosc/128
“100”	fosc/256
“101”	fosc/512
“110”	fosc/1024
“111”	fosc/2048

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
SPSTAT	SPIF	WCOL	PSC2	--	--	--	--	--

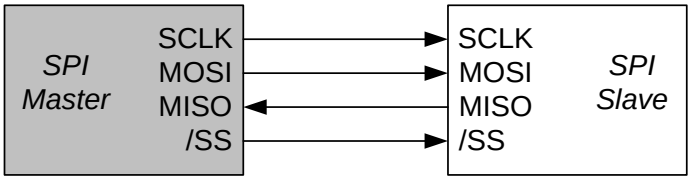
- SPIF: SPI0 transfer completion flag. When ESPI (IE.5) bit and EA (IE.7) bit are '1' and a SPI0 transfer finishes then the SPIF bit is set by hardware and an interrupt is generated. SPIF will also be set when SPI0 is in master mode with SSIG(SPCTL.7)='0' and /SS0(slave select) pin is driven low
- WCOL: SPI0 write collision flag. The WCOL bit is set by hardware when SPDAT is written during a data transfer. WCOL flag is cleared in software by writing '0' to this bit
- PSC2: SPI0 master clock rate select msb
- --: Reserved bit

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
SPDAT	B7	B6	B5	B4	B3	B2	B1	B0

- B(7:0): Data transferred. B7 msb; B0 lsb.

The SPI interface requires four pins: SPICLK, MOSI, MISO and SS:

Figure 8-1. SPI connection diagram



SPI0 pins are mapped to P5 port pins as alternate output functions

Table 8-1. SPI0 port map

ADD8051C3A Port	Alternate Function
P5.0	/SS0
P5.1	SPICLK0
P5.2	MOSI0
P5.3	MISO0

SPICLK, MOSI and MISO are typically tied together between two or more SPI devices. Data flows from master to slave on the MOSI (Master Out Slave In) pin and flows from slave to master on the MISO (Master In Slave Out) pin. The SPICLK signal is output in the master mode and is input in the slave mode. If the SPI system is disabled, i.e. SPEN (SPCTL.6) = 0 (reset value), these pins can be used as general purpose I/O pins.

SS is an optional slave select pin. In a typical configuration, an SPI master asserts one of its port pins to select one SPI device as the current slave. An SPI slave device uses its SS pin to determine whether it is selected or not. The SS is ignored if any of the following conditions are true:

- The SPI0 system is disabled, i.e. SPEN (SPCTL.6) = 0 (reset value)
- SPI0 is enabled but SS pin is not needed in such system, i.e. SSIG(SPCTL.7) = 1; in this case SS pin can be used as general purpose pin on P5.

8.1.2 SPI clock phase, polarity and operation

There are four combinations (CPHA, CPOL) for sampling and shifting activities on data and clock lines in the SPI. Master can switch between any of them at any time thus having ability to communicate with slaves supporting data transfer using different modes.

In order to have successful data transfer between SPI devices, proper selection of the SPI clock phase and polarity is crucial. It is important to note that some of these configurations offer less capabilities than others. Clock Phase Bit CPHA allows user to specify the edges for sampling and shifting data. Clock Polarity bit CPOL allows user to set the clock polarity Figure 8-2 and Figure 8-3 show transfers with different values of CPHA and CPOL.

The SPI clock preescaler selection uses the PSC1-PSC0 bits in the SPCTL register and PSC2 in SPSTAT register. Master selects one of the available baud rates for the SPI communication. SPI Clock Preescaler bits do not have affect on the part acting as a slave, since it uses SPI clock supplied by the master.

When microcontroller operates as a slave with CPHA=’0’, some restrictions are present.

- SSIG must be ‘0’ and the SS pin must be negated and reasserted between each successive byte transfer.
- If the SPDAT register is written while SS is active (low), a write collision error results.
- Microcontroller’s behavior is undefined if CPHA is ‘0’ and SSIG is ‘1’.

On the other hand, slave having CPHA=’1’ may set SSIG to ‘1’. If SSIG = 1, the SS pin may remain active low between successive transfers (can be tied low at all times). This format is sometimes preferred in systems having a single fixed master and a single slave driving the MISO data line. Microcontroller configured as a master with CPHA=’0’ does not need to negate and reassert slave’s SS line in order to send and receive byte(s) of data.

In SPI, transfers are always initiated by the master. If the SPI is enabled (SPEN = 1) and microcontroller is configured as SPI master, writing to the SPI data register by the master will start the SPI clock generator and data transfer. The data will start to appear on MOSI about one half SPI bit-time to one SPI bit-time after data is written to SPDAT.

Figure 8-2. SPI0 transfer with CPHA=0

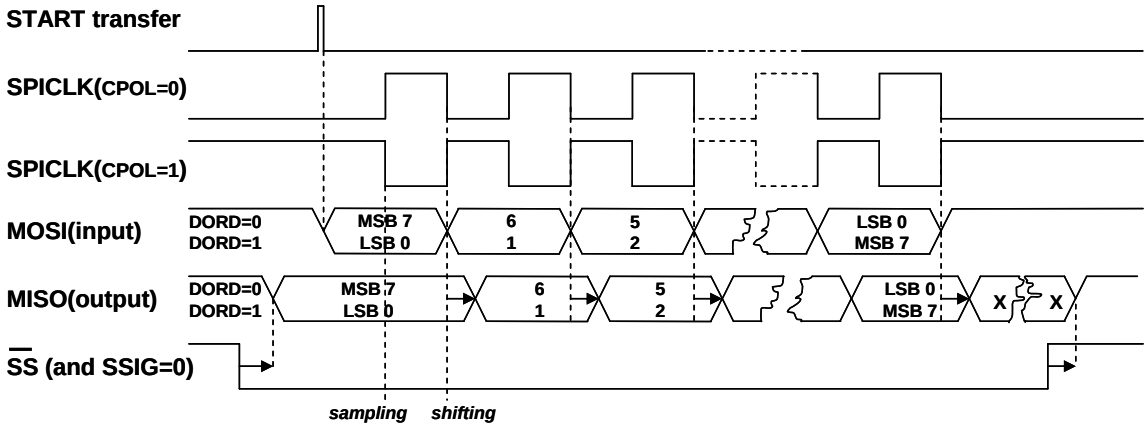
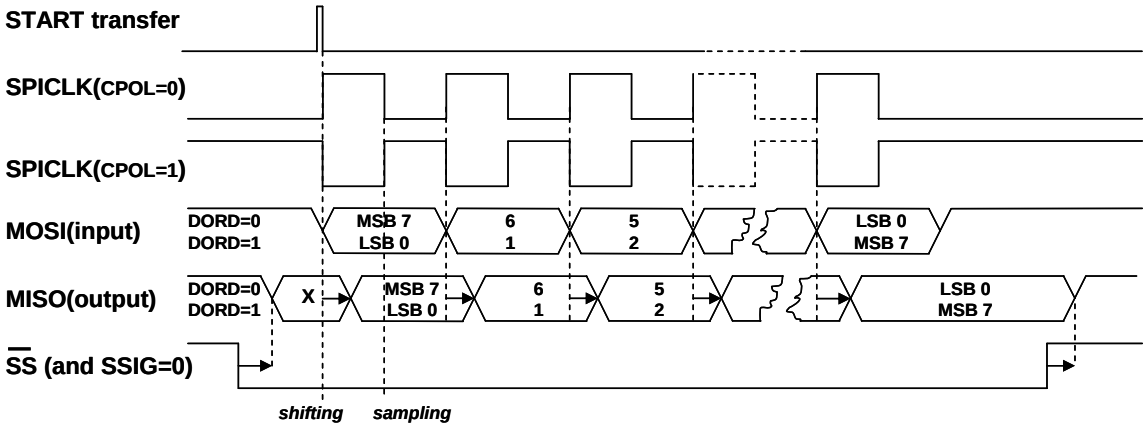


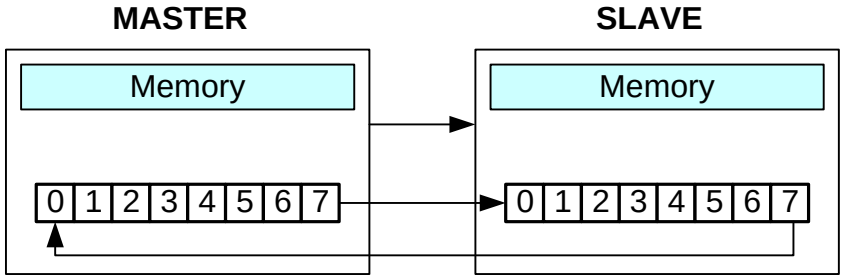
Figure 8-3. SPI0 transfer with CPHA=1



Note that the master selects a slave by driving the slave select pin of the corresponding slave device. Data written to the SPDAT register of the master is shifted out of the MOSI pin of the master to the MOSI pin of the slave, at the same time the data in SPDAT register on slave side is shifted out on its MISO pin to the MISO pin of the master.

After shifting one byte, the SPI clock generator stops, setting the transfer completion flag (SPIF) and an interrupt will be created if the SPI interrupt is enabled (ESPI, or IEN1.3 = 1). The two shift registers in the master CPU and slave CPU can be considered as one distributed 16-bit circular shift register. When data is shifted from the master to the slave, data is also shifted in the opposite direction simultaneously. This means that during one shift cycle, data in the master and the slave are interchanged.

Figure 8-4. SPI shift registers



If SPEN = 1, SSIG = 0 and MSTR = 1, the SPI0 is enabled in master mode. The SS pin is quasi-bidirectional. In this case, another master can drive this pin low to select this device as an SPI slave and start sending data to it. To avoid bus contention, the CPU becomes a slave. As a result of the observed SPI module becoming a slave, the MOSI and SPICLK pins are forced to be an input and MISO becomes an output.

The SPIF flag in SPSTAT is set, and if the SPI interrupt is enabled an SPI interrupt will occur.

User software should always check the MSTR bit. If this bit is cleared by a slave select and the user wants to continue to use the SPI as a master, the user must set the MSTR bit, otherwise it will stay in slave mode.

8.1.3 SPI0 write collision

The SPI0 is single buffered in the transmit direction and double buffered in the receive direction. New data for transmission cannot be written to the shift register until the previous transaction is completed. The WCOL (SPSTAT.6) bit is set to indicate data collision when the data register is written during transmission. In this case, the data currently being transmitted will continue to be transmitted, but the new data, i.e., the one causing the collision, will be lost.

While write collision is detected for either a master or a slave, it is uncommon for a master because the master has full control of the transfer in progress. The slave, however, has no control over transmission start by the master and therefore collision can occur.

Receiver transfers received data into a parallel read data buffer so that the shift register is free to accept a second character. However, the received character must be read from the Data Register before the next character has been completely shifted in. Otherwise. the previous data is lost. WCOL can be cleared in software by writing a ‘0’ to this bit.

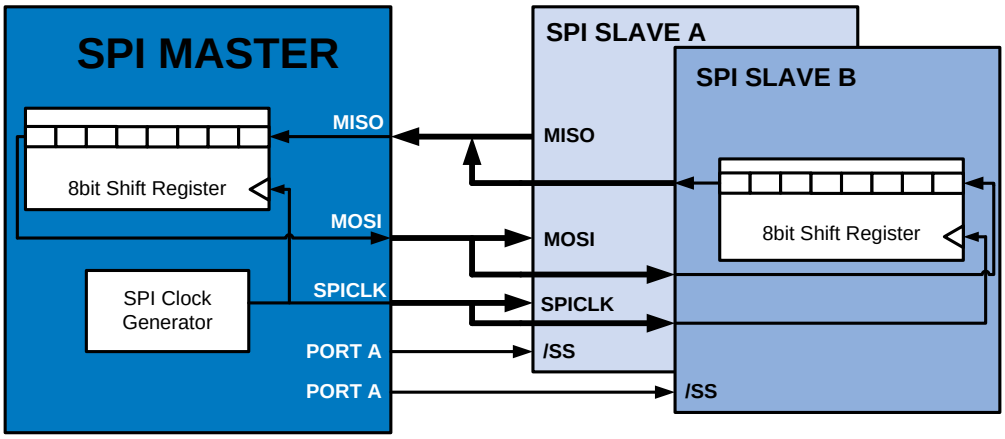
8.2 SPI Modes

The communication using SPI in a single-master system is simple and usually works as described below:

- Both master and slave(s) configure their SPIs to operate in the same mode (one of four available modes) and turn them on; additionally, the master has to select baud rate for the communication; slave uses master’s serial clock to sample and shift data during the transfer
- Master selects desired slave unit(s) using its/their SS line(s)
- As soon as the master writes to its SPI buffer register (assuming that slave(s) has/have already loaded data into its/their buffer(s)), transfer is initiated; master’s SPI module generates serial clock and master’s and slave’s data are exchanged
- After the end of transfer, an indication is generated in all participating SPI units, both the master and the slave(s) read received data from their buffers
- If there is more data to be exchanged, all units taking part in this communication prepare the new set of data and master initiates another round of transfer; if there is no more data to be exchanged, master deselects used SS line(s) and previously active slaves are deselected; this is the end of SPI transfer

Typical connection in system using SPI is shown below.

Figure 8-5. SPI single master with multiple independent slaves configuration



In Figure 8-5, slave's SSIG = 0 (see 8.1), and SS is used to select the slave. The SPI0 master can use any port pin (including P5.0/SS with SSIG = 1) to drive the SS pin.

Multimaster configuration is the case where two devices are connected to each other and either device can be a master or a slave. When no SPI0 operation is occurring, both can be configured as masters (MSTR = 1) with SSIG = 0 and P5.0 (SS) being in quasi-bidirectional mode. Before device initiates a transfer, it must set SSIG = 1 in order to avoid its own mode change since it will drive P5.0 low, forcing a mode change in the other device to slave.

Table 8-2 shows configuration for the master/slave modes in various cases.

Table 8-2. SPI modes

SPEN	SSIG	SS pin	MSTR	operation	MISO	MOSI	SPICLK	comment
0	X	P5.0(1)	X	disabled	P5.3(1)	P5.2(1)	P5.1(1)	SPI0 disabled, P5(3:0) are used as port pins
1	0	0	0	slave	out	in	in	selected as slave
1	0	1	0	slave	H	in	in	not selected, MISO is H to avoid bus contention
1	0	0	1(=>0) (2)	slave	out	in	in	the uC is configured as a master, then it is selected by SS to be a slave, MSTR bit is cleared and the uC becomes a slave
1	0	1	1	master	in	out	out	
1	1	P5.0(1)	0	slave	out	in	in	slave not needing SS
1	1	P5.0(1)	1	master	in	H	H	master not needing SS

Notes: 1. function as port or as alternate output function
2. the MSTR bit changes to '0' automatically when SS becomes low in input mode and SSIG is 0

8.3 SPI1 buffer operation

SPI1 is a byte buffered serial peripheral interface. Data transfers have a selectable additional number of bytes ranging from 0 to 7, SPSTAT1(2:0) is used to select the number of additional bytes. The buffer is composed of the registers SPDAT10 to SPDAT17. The content of SPDAT10 is the first to be transferred.

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
SPCTL1	SSIG	SPEN	DORD	MSTR	CPOL	CPHA	PSC1	PSC0

- SSIG: /SS1 slave select ignore, if set to '1' MSTR decides whether the device is a master or a slave and /SS1 pin can be used as port pin; if set to '0' and MSTR is set to '1' then the /SS1 pin decides whether the device is a master or a slave
- SPEN: SPI1 enabled '1' or disabled '0', when the SPI1 is disabled the SPI1 pins can be used as general I/O pins
- DORD: SPI1 data order

  '1' - the lsb of the data byte is transmitted first

  '0' - the msb of the data byte is transmitted first
- MSTR: master '1' / slave '0' mode select. See SSIG field
- CPOL: SPI1 clock polarity

  '1' – SPICLK1 is high when idle, the leading edge of SPICLK1 is the falling edge and the trailing edge is the rising edge

  '0' – SPICLK1 is low when idle, the leading edge of SPICLK1 is the rising edge and the trailing edge is the falling edge
- CPHA: SPI1 clock phase select

  '1' - data is driven on the leading edge of SPICLK1 and is sampled on the trailing edge

  '0' - the first data bit is on the MOSI1/MISO1 line before the first SPICLK1 leading edge, data is sampled on the leading edge of SPICLK1 and driven on the trailing edge of SPICLK1
- PSC(1:0): SPI1 clock rate select, determines master clock output. 8 different baud rates can be selected using PSC2[SPSTAT1(5)] & PSC(1:0)[SPCTL1(1:0)]

  When working as slave, PSC(2:0) value is not taken into account, nevertheless this value must be different from “000”

PSC(2:0)	**clock rate**
“000”	fosc/4
“001”	fosc/16
“010”	fosc/64
“011”	fosc/128
“100”	fosc/256
“101”	fosc/512
“110”	fosc/1024

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
SPSTAT1	SPIF	WCOL	PSC2	--	--	N2	N1	N0

- **--:** Reserved bit
- **SPIF:** SPI1 transfer completion flag. When ESPI1 (IE2.3) bit and EA (IE.7) bit are '1' and a SPI1 transfer finishes then the SPIF bit is set by hardware and an interrupt is generated. SPIF will also be set when SPI1 is in master mode with SSIG(SPCTL1.7)='0' and /SS(slave select) pin is driven low
- **WCOL:** SPI1 write collision flag. The WCOL bit is set by hardware when SPDAT is written during a data transfer. WCOL flag is cleared in software by writing '0' to this bit.
- **PSC2:** SPI1 master clock rate select msb
- **N(2:0):** Number of additional bytes to transfer in SPI1

The data bytes to be transferred must be written in order. Transfer will start when SPDAT1N is written if N=SPSTAT(2:0).

A write collision will occur if any SPDAT1N (with N<=SPSTAT(2:0)) is written before the 1+N bytes transfer is finished.

Table 8-3. SPI1 port map

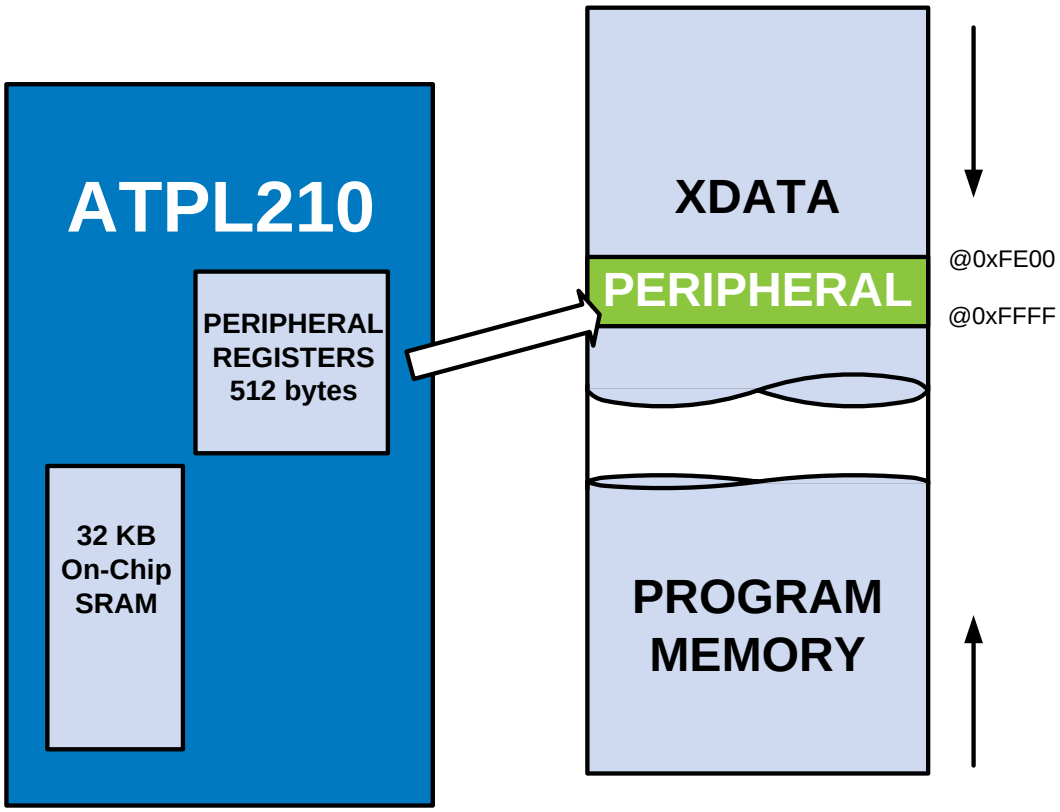
ADD8051C3A Port	Alternate Function
P4.2	/SS1
P4.3	SPICLK1
P4.4	MOSI1
P4.5	MISO1

9. Peripheral Registers

A 512 bytes space is reserved on-chip to allocate the system peripheral registers. These registers are mapped in XDATA (External Data Memory space) in addresses from FE00 to FFFF, so these addresses cannot be used to store user data. This fact must be taken into account by the user when calculating the space needed to store the application data.

On-chip Peripheral Registers are mapped in External Data Memory. They can be accessed addressing FE00 to FFFF.

Figure 9-1. ATPL210 Peripheral Register mapped in Memory

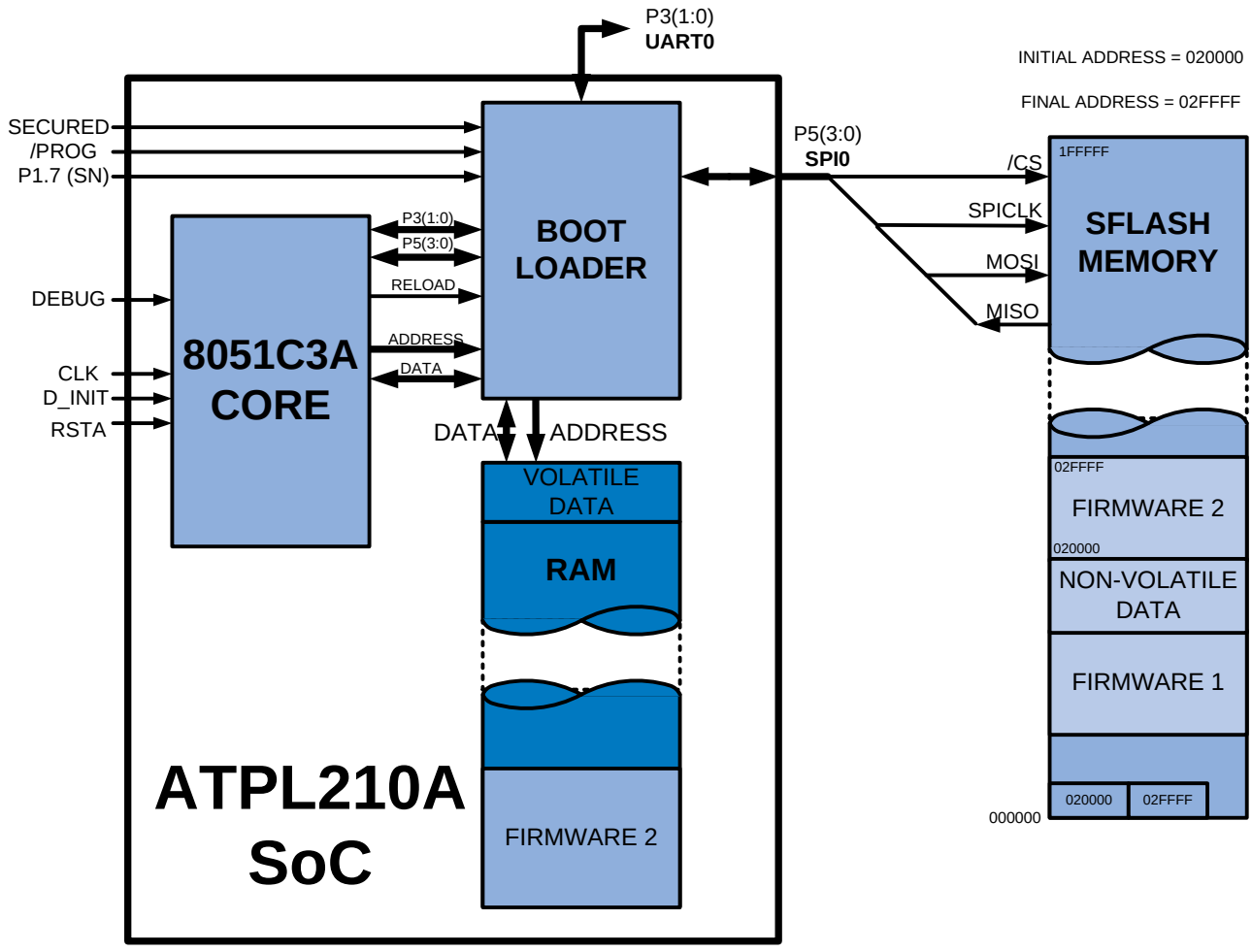


10. Boot Loader

ATPL210A needs an external SPI flash memory to store the firmware and all no-volatile data, while an SRAM (internal or external) is used for firmware execution (once a volatile firmware copy has been automatically generated by the boot loader in the system startup process) and to store all the volatile variables.

After power-up the boot loader forces a start-up cycle uploading the target program from the serial flash to the SRAM, the program is executed from the lower entries of the SRAM and the upper entries are used to store program data. A boot loader diagram is shown in Figure 10-1

Figure 10-1. Boot-loader diagram



Boot loader is executed in the following situations:

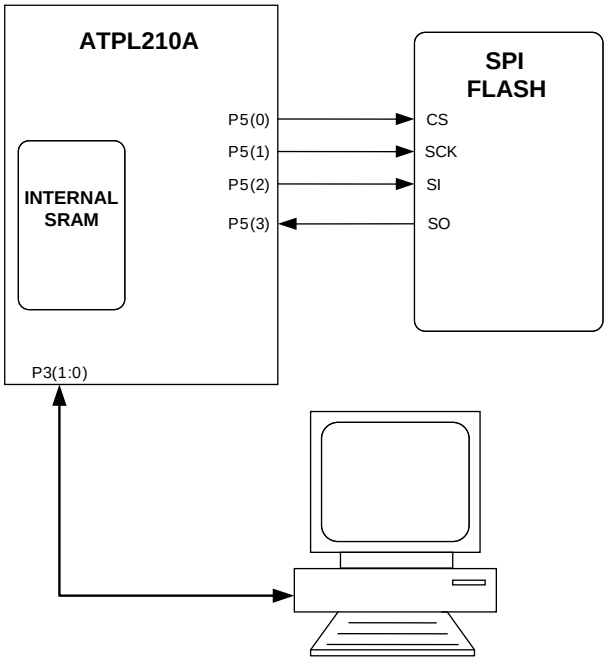
- Every time after a Power-On.
- Every time the system is reset by the asynchronous reset signal (RSTA).
- Every time the bit 2 in CONF SFR is set, forcing a reload.
- After a Watchdog timeout, only if bit 1 in CONF SFR is set.

The boot loader circuitry also supports in-system programming. A user can update the firmware via a serial interface without additional logic and without disassembling the system.

When a SPI flash with serial number or/and a Silicon Serial Number IC are available in the system board, the system supports automatic firmware encryption in the flash.

In the ATPL210A the microcontroller can force program reloading by writing in CONF Specific Function Register.

Figure 10-2. Flash connection diagram



10.2 Pin Description

- **/PROG:** After power up, if this signal is tied low the in-system programming mode is enabled, else the system is in execution mode.
  - '0': In-system programming mode
  - '1': Execution mode
- **SECURED:** This pin enables encrypted firmware execution when the board configuration supports it (see Encrypted firmware requirements section).
  - '0': Cryptographic Storage disabled
  - '1': Cryptographic Storage enabled
- **P1.7 (SSN):** Input pin used to read a Serial Number if a valid Silicon Serial Number device is being used (see 10.4.2)

As shown in Figure 10-1 the boot loader acts as a bus switcher. It takes control over the Standard Serial port 0 (P3[1:0]) and pins of port P5 (P5[3:0]) that are used as Serial Peripheral Interface to connect the serial flash (SPI0). See table below:

Table 10-1. SS0 & SPI0 buses

ADD8051C3A Port	Alternate Function
P3.0	RxD0
P3.1	TxD0
P5.0	/SS0
P5.1	SPICLK0
P5.1	MOSI0
P5.3	MISO0

After program booting, the shared pins are fully software controlled. This allows the microcontroller to store no-volatile program data to the serial flash. The microcontroller has full access to all the serial flash content which **has to be handled carefully to avoid unexpected overwriting of program code located in the flash memory.**

10.3 Flash Programming

10.3.1 In-System programming

The boot loader supports in-system programming for SPI flash memories.

To enable in-system programming /PROG pin must be tied low.

When in-system programming is enabled, the boot loader is controlled via RS-232 commands.

The default RS-232 configuration is 57600 bauds, 8 data bits, 1 stop bit and no parity.

Atmel provides a set of tools that allows customers to carry on in-system programming. For further information, please contact Atmel support.

10.3.2 SPI Flash programming

After a RESET, if /PROG pin is tied low, SPI flash commands are allowed. This set of commands allows the user to send any standard SPI command to the flash memory through the serial port. The boot loader acts as a bridge from serial to SPI protocol.

The serial commands are composed by a header for the boot loader and the SPI command itself.

Figure 10-3. Serial commands

- **ICODE:** This byte indicates to the boot loader the type of instruction to be executed

ICODE (Hex)	RS-232 FORMAT (Hex)
01	01 N-1 OPC AD2 AD1 AD0 D1 DN
02	02 N-1 OPC AD2 AD1 AD0
04	04 XX OPC
08	08 XX OPC AD2 AD1 AD0
10	10 N-1 OPC
20	20 N-1 OPC D1 DN

- **BC:** Byte Count indicates the number of data bytes to read or write. The value of this field must be the number of data bytes to read or write minus one, that is, the range of the BC field is [0,255] and the number of data bytes range is [1,256]. If the requested instruction is not a reading or writing instruction the BC field is not used and can take any value.

Note: If Atmel SPI flash memory (see 10.4.3) is being used, and it's configured with a page size of 264 bytes, BC field will have a length of 2 bytes.

- **ADDR(2:0):** 24 bits address field. This field must point to the initial address of the instruction operation.
  DATA(1:n): Data bytes of the SPI instruction (n = BC + 1). The DATA1 byte is the first data byte in a write or read operation.

With the six command types listed in ICODE description, it is possible to execute any standard operation on the SPI flash memory:

Table 10-2. ICODE operations

ICODE (Hex)	Typical Use
01	Page programming
02	Read, Read id.
04	Write enable/disable
08	Erase
10	Read status register
20	Write status register

Example: Programming 256 bytes to flash memory starting at 010000 (Hex) address. (The page programming opcode of the memory is 02 (Hex) and the required erase procedure is not included in this example).

The RS-232 command in hexadecimal format is:

01.FF.02.01.00.00.[byte1 to byte256]

10.4 System startup

When the system is configured in execution mode (/PROG pin is set) and after a power up or reload process, the boot loader checks the system configuration and performs the required operations to ensure the correct execution of the firmware.

Boot loader transfers a volatile copy of the firmware from the flash memory to the SRAM memory starting at address 000000 (Hex). The maximum SPI flash size supported is 16Mbits. The part of the SRAM that is not used to store the firmware is used by the system to store the volatile data in the execution process.

The volatile data is stored in the SRAM memory from top to bottom. This arrangement is done automatically by the system, that is, when the microcontroller wants to save data at address A the system automatically converts the target address to (top address – A). Using this storage method, the amount of volatile data that can be stored in the RAM is determined by the firmware size.

The transfer process can last about 400ms depending on flash device and code size. The first time this process will be longer if auto encryption is activated. Auto encryption must be done once, and **care must be taken about respecting encryption times**.

When the transfer process is finished, the microcontroller begins to execute the program and the system is ready to use. The startup cycle is performed each time the system is powered up or when the microcontroller via software or watchdog forces a firmware reload process writing in a specific register.

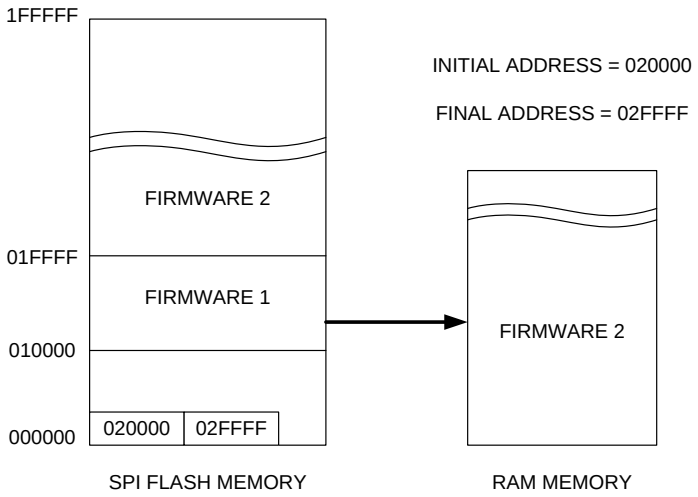
Both the location of the firmware at the SPI flash and the transfer size are configurable. The lower six bytes of the SPI flash are used to store the initial and final position of the firmware (from address 0x000000 to 0x000005).

This feature allows working with a SPI flash device that has multiple firmware versions stored in different addresses. The version to be used can be selected by means of its initial and final addresses.

The transfer size is equal to the firmware size.

Bank-switching is supported, so firmware code bigger than 64KB can be managed. Extreme caution must be taken when using bank-switching in order to respect code and variables spaces and avoid overlapping.

Figure 10-4. Flash & SRAM memories diagram



10.4.2 Encrypted firmware requirements

In order to run Auto-encryption procedure, the conditions below must be met:

- SECURED = ‘1’
- DEBUG = ‘0’
- /PROG = ‘1’
- Valid Serial Number (see supported devices section)
- Correct flash memory START and END addresses
- Detection by hardware of no-encrypted software stored in a valid flash device
- Starting address of the stored firmware multiple of 4KB (0xXXX000)

First time auto-encryption procedure runs, the code is encrypted and rounded to the next multiple of 4kbytes, and the executions begins.

Next time the system boots, hardware will detect encrypted software in the flash, thus auto encryption procedure will not be started (that is, auto encryption is only necessary to be performed once).

Note1: Auto encryption procedure takes about 30 seconds to be completed, and during encryption no external signals or LEDs are asserted, so care must be taken by the user and encryption time must be respected.

Note2: If SECURED = ‘0’ or some of the other conditions above are not met, the system may run without code encryption. Care must be taken by the user.

Note3: If SECURED = ‘0’ and the code stored in the flash is encrypted (an unusual situation), the system will be expecting no-encrypted code and will fail.

Note 4: A helpful way to check that the firmware in the flash has been encrypted, is to read the start and end addresses allocated at the beginning of the code. If encryption has been done, these address values will make no sense because of the encryption.

10.4.3 Supported Devices

The boot loader supports any SPI flash memory compatible with SPI mode 0 (CLK signal is normally low) and with a value of read instruction operation code equals to 0x03 (Hex). The maximum supported flash size is 16Mbits

Atmel AT45DBxx SPI Flash family is strongly recommended and has been fully tested in ATPL210A development kits and in field configurations.

10.4.3.1 Serial Number Device

If the system is configured to use encrypted firmware it is necessary to include in the system board a device that provides a serial number to the ATPL210A SoC.

Atmel Flash AT45DBxx device also includes its own Serial Number, so the flash device provides the Serial Number itself and there is not necessary an external SN chip.

When a more robust encryption is desired, a SN device can be added to the system, for example:

- DS2401 (Dallas): IC Silicon Serial Number. Provides a valid serial number to be used for encryption purposes.

When both devices are used together (AT45DBxx+DS2401), a more robust encryption is achieved.

10.5 Boot loader registers

10.5.1 SN_BYTE registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
SN_BYTE	SN_BYTE(63:56)								@0xFE4B
	…							…	
	SN_BYTE(7:0)								@0xFE52

Name: SN_BYTE

Address: 0xFE4B – 0xFE52

Access: Read only

Reset: 0x00, …, 0x00

These registers store a 64-bit Silicon Serial Number used in auto-encryption. This serial number is provided by an external device and is only necessary if the system uses auto-encryption features.

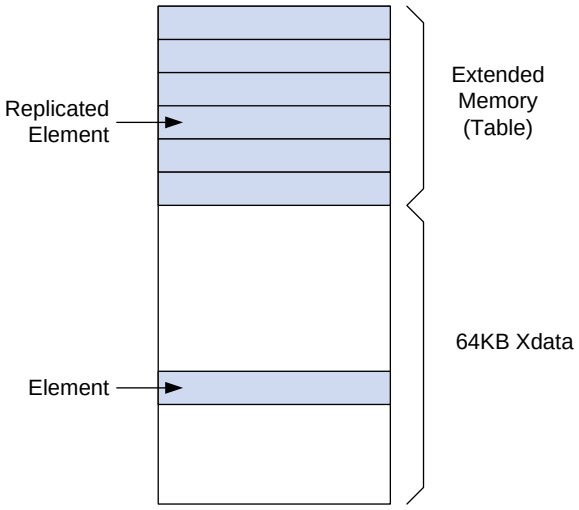
11. Data Memory Extension Block

In ADD8051C3A microcontroller data memory consists of 64 Kbytes of external memory. Due to data table handling in software development for PRIME MAC implementation, it could be useful to extend data memory above 64 Kbytes. For this extension a specific hardware peripheral block has been implemented, with the functionality of address regenerating when needed.

11.1 Description

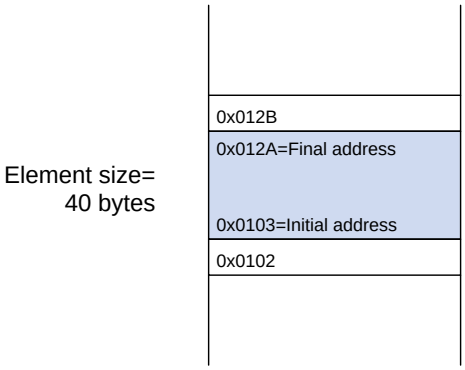
This block consists of an address extension when address is located within specific limits. The new address is calculated using specific registers, providing capability of replicating a limited space in the data memory so many times as possible with the memory resources and addressing capacity. The space in data memory to be extended is defined by the initial address and the size of this space. This memory space could correspond to an element of a table (data structure), so it would be possible to define a table with a high number of these elements. In order to change from an element to another it is defined the index of the element, used for generating the corresponding address.

Figure 11-1. Basic Memory Extension



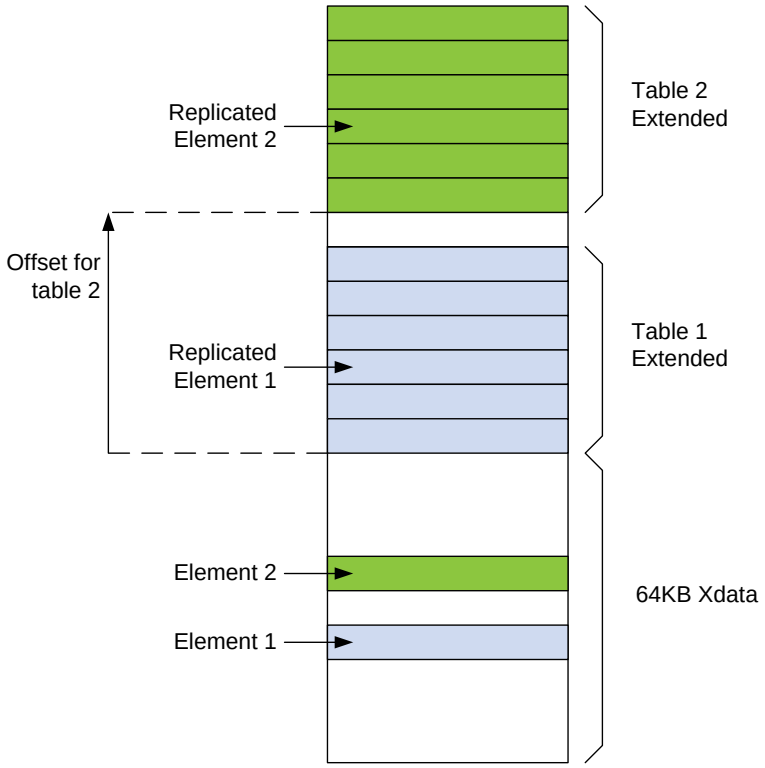
In Figure 11-1 it is shown the basic concept about memory extension, which it is thought as an element-replication at high positions of the data memory. The first replicated element starts just after 64 KB from xdata, and the next elements are located at following positions in memory. The element has a configurable size, with a maximum size of 1 KB. If the size of the element is set to 0, then this hardware block is disabled and no extension is applied.

Figure 11-2. Element Example



It is also possible to define several tables, so it is needed to define different initial addresses and sizes, from different spaces (elements of the tables). In this case the extended memory also requires an address offset to change from a table to another.

Figure 11-3. Two tables with Memory extension



When using several tables, it is needed to change the offset value, the initial address of the corresponding element, the element size if they are different and the index of the element.

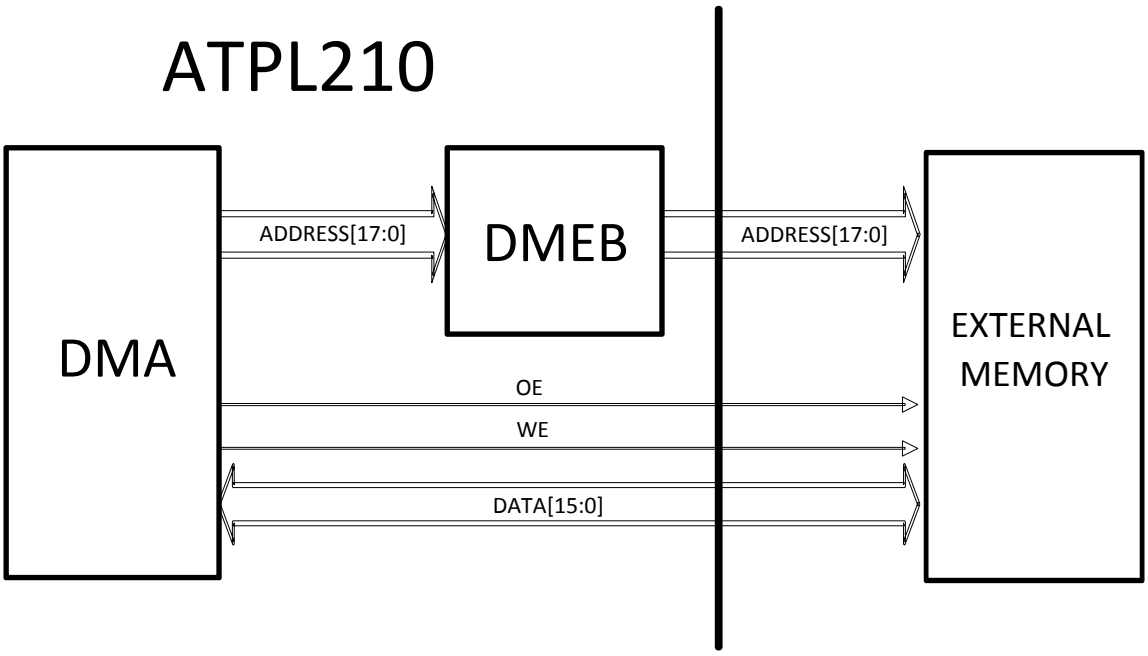
If it is trying to access to the external memory at the corresponding positions to one of this elements, then the hardware block calculates a new address according to its parameters, as follows:

NEW_ADDRESS = 0x10000 + (ADDRESS – INITIAL_ADDRESS) + ELEMENT_SIZE * ELEMENT_INDEX + OFFSET

where ADDRESS is a valid address higher or equal than the initial address of the element and lower than the final address (initial address+element size). INITIAL_ADDRESS is obviously the initial address of the element, ELEMENT_SIZE is the size of the element, ELEMENT_INDEX is its index in the table, OFFSET is the address offset to change among tables, 0x10000 is the first address available above 64 KB from xdata and NEW_ADDRESS is the new address calculated and multiplexed by this block.

The function of this block is to multiplex the address when accessing external data memory, so the rest of ram data and control signals do not take part in data memory extension.

Figure 11-4. Data Memory Extension Block (DMEB) in ATPL210



It is needed to consider that program memory is also included in external memory, so both program and data memories of ADD8051C3A share the total RAM size, thus the data memory extension capability depends also on program memory size, in order to avoid any overwriting between these spaces.

11.2 DMEB Configuration Registers

Table 11-1 shows the configuration peripheral registers involved in Data Memory Extension block:

Table 11-1. Data Memory Extension Block registers address map

Address	Register Name	Description
0xFE95	TABLE_ELEMENT_SIZE	See 11.2.2
0xFE96		
0xFE97	TABLE_INDEX	See 11.2.3
0xFE98		
0xFE99	TABLE_ELEMENT_INIT	See 11.2.4
0xFE9A		
0xFEA7	TABLE_OFFSET	See 11.2.5
0xFEA8		
0xFEA9		
0xFEAA		

As explained in Peripheral Registers, Peripheral Registers are mapped in XDATA (External Data Memory space) in addresses from FE00 to FFFF.

11.2.2 TABLE_ELEMENT_SIZE registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
TABLE_ELEMENT_SIZE	--						TABLE_ELEMENT_SIZE(9:8)		@0xFE95
	TABLE_ELEMENT_SIZE(7:0)								@0xFE96

Name: TABLE_ELEMENT_SIZE

Address: 0xFE95 – 0xFE96

Access: Read/write

Reset: 0x00, 0x00

- **TABLE_ELEMENT_SIZE:** Size in bytes of the element of the table in extended data memory. The element with this size can be replicated so many times as needed within the available memory. It is important to know the number of elements in the table, in order to verify if the table exceeds memory resources. Note that RAM is used by both program and data memory, therefore carefulness must be taken by the user in order to not overwrite the program memory with the higher elements of the table.

11.2.3 TABLE_INDEX registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
TABLE_INDEX	--						TABLE_INDEX(9:8)		@0xFE97
	TABLE_INDEX(7:0)								@0xFE98

Name: TABLE_INDEX

Address: 0xFE97 – 0xFE98

Access: Read/write

Reset: 0x00, 0x00

- **TABLE_INDEX:** Index of the element of the table in extended data memory. The position memory to be read will be:

65536+TABLE_INDEX*TABLE_ELEMENT_SIZE+(ADDRESS- TABLE_ELEMENT_INIT) + TABLE_ OFFSET

where ADDRESS is the position at XDATA memory to be read of the microcontroller.

11.2.4 TABLE_ELEMENT_INIT registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
TABLE_ELEMENT_INIT	TABLE_ELEMENT_INIT(15:8)								@0xFE99
	TABLE_ELEMENT_INIT(7:0)								@0xFE9A

Name: TABLE_ELEMENT_INIT

Address: 0xFE99 – 0xFE9A

Access: Read/write

Reset: 0x00, 0x00

- **TABLE_ELEMENT_INIT:** Initial address of the table in data memory. This is the address of the element at the XDATA memory of the microcontroller (64KB-space). Positions from this initial address to TABLE_ELEMENT_INIT+TABLE_ELEMENT_SIZE-1 will never be accessed when data memory extension is used.

11.2.5 TABLE_OFFSET registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
TABLE_OFFSET	--								@0xFEA7
	--			TABLE_OFFSET(20:16)					@0xFEA8
	TABLE_OFFSET(15:8)								@0xFEA9
	TABLE_OFFSET(7:0)								@0xFEAA

Name: TABLE_OFFSET

Address: 0xFEA7 – 0xFEAA

Access: Read/write

Reset: 0x00;…; 0x00

- **TABLE_OFFSET:** 21-bit offset of the table in extended data memory

11.3 Code Example

Following, a brief C code programming example of DMEBwill be explained.

Please note that the total number of elements is not defined. The maximum number of elements is fixed by memory total size. Thus, extra care must be taken by the user in order to not overwrite the program memory with the higher elements of the table.

Pointers must not be used as table elements, to avoid conflict between hardware and compiler.

ONE TABLE
---

1. *Table definition (struct) in include *.h*
   Element size = 24 bytes; Uint8 = unsigned integer, 8 bits; Uint16 = unsigned integer, 16bits.

```
struct ExampleTable
        {
                Uint8 Element0;
                Uint8 Element1;
                Uint16 Element2;
                Uint8 Vector[20];
        }
```

2. *Element declaration as a variable in source file *.c, occupying 24 bytes in XDATA.*
   "_at_" means the allocation of the variable in a concrete position in data memory.

```
 struct ExampleTable TableElement _at_ 0x0200;
```

3. *Initialization of DMEB to use a table with this element replicated out of the XDATA 64KB.*

   Variables TABLE_ELEMENT_SIZE, TABLE_ELEMENT_INIT and TABLE_INDEX are the DMEB peripheral registers.

```
TABLE_ELEMENT_SIZE = sizeof (TableElement);
TABLE_ELEMENT_INIT = (Uint16)&TableElement;
TABLE_INDEX = 1;
TABLE_OFFSET = 0;
```

4. *Table management: reading/writing element 12. Only one DMEB register must be modified when changing between elements, TABLE_INDEX.*

```
TABLE_INDEX = 12;
        if (TableElement.Element0)
        {
                TableElement.Element1 = 0x46;
                TableElement.Element2 = 0x1FB2;
        }
        TableElement.Vector[0] = 0x04;
        TableElement.Vector[1] = 0x05;
...
```

MORE THAN ONE TABLE. TABLES WITH THE SAME NUMBER OF ELEMENTS

---

1. *Another struct is defined. New struct size is 3 bytes.*

```
struct ExampleTableAdditional
        {
                Uint8 Element0;
                Uint16 Element1;
        }
```

2. *Element declaration in an XDATA address. In this case, both elements are declared in consecutive memory locations. Thus the hardware will know that both elements are together, and it will set the element size equal to the addition of the different type elements.*

```
struct   ExampleTable TableElement _at_ 0x0200;
struct   ExampleTableAdditional TableElementAdditional _at_ 0x0200+ sizeof (TableElement );
```

3. *Initialization of DMEB to use both tables.*

```
TABLE_ELEMENT_SIZE = sizeof (TableElement) + sizeof (TableElementAdditional);
TABLE_ELEMENT_INIT = (Uint16)&TableElement;
TABLE_INDEX = 1;
TABLE_OFFSET = 0;
```

4. *Table management: Reading element 5 from TableElement, writing element to index 10 in TableElementAdditional. Only one DMEB register must be modified when changing between elements, TABLE_INDEX.*

```
TABLE_INDEX = 5;
aux = TableElement.Element0;
TABLE_INDEX = 10;
TableElementAdditional.Element0 = aux;
...
```

MORE THAN ONE TABLE. TABLES WITH DIFFERENT NUMBER OF ELEMENTS

---

1. *With the structures defined above, also it is possible to use two tables with different number of elements. In this case, to realize an efficient memory usage, is useful to calculate the memory size of each table, and then work with an offset for each one. If the element differs from one table to another, then it will be necessary to declare variables in different locations and also set the element size (TABLE_ELEMENT_SIZE and TABLE_OFFSET) every time a table change occurs.*

```
struct ExampleTable TableElement _at_ 0x0200;
struct ExampleTableAdditional TableElementAdditional _at_ 0x0300;
```

2. *Initialization of one of the tables.*

```
TABLE_ELEMENT_SIZE = sizeof (TableElement);
TABLE_ELEMENT_INIT = (Uint16)&TableElement;
TABLE_INDEX = 1;
TABLE_OFFSET = 0;
```

3. *Table management: reading element 4 in TableElement and writing element 7 in TableElementAdditional. In this case, every time a table change occurs, the following values must be changed: element size, element initial address, table offset, element index.*

```
TABLE_INDEX = 4;
aux = TableElement.Element0;
TABLE_ELEMENT_SIZE = sizeof (TableElementAdditional);
TABLE_ELEMENT_INIT = (Uint16)&TableElementAdditional;
TABLE_OFFSET = numElements * sizeof (TableElement);
TABLE_INDEX = 7
TableElementAdditional.Element0 = aux;
```

12. Direct Memory Access (DMA)

ATPL210A SoC includes 4 DMA (Direct Memory Access) channels which provide an enhanced way for memory transference between positions from the same data memory location or between peripherals and the external data memory. The main goal is to increase memory transference performance, comparing with ADD8051C3A microcontroller using MOVX instructions. Two of these DMA channels are used by the physical layer and are not visible by the microcontroller or another interface. These two channels (PHY DMA channels) have a similar structure to the other two channels (generic DMA channels), but they are not configurable, except addresses –destination in reception and source in transmission-.

The other two DMA channels are generic and have some configurable parameters: source and destination addresses, source and destination sizes, time of each DMA access and time between consecutive accesses

12.1 DMA Management

DMA channels are intended to be used by means of a user interface.

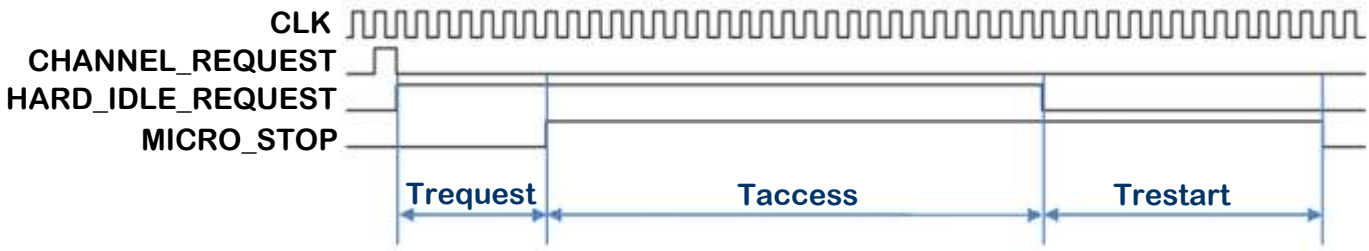
By using a software interface, the user can perform DMA write/read operations and DMA configuration, so low-level management becomes transparent. Please see “PRIME User Help” for further information about software management.

12.2 DMA Hardware

DMA requires to access data memory, so the microcontroller execution must be stopped. It is needed a Hard Idle ReQuest (HIRQ) to the microcontroller, in order to save the current state with the current instruction complete. There is a variable time from the DMA request to the micro stop, depending on its current execution state. When the microcontroller is correctly stopped a signal is provided to DMA to get full access to data memory. When transference is completed, the hard idle request is cleared and the microcontroller shall resume its execution. The signal provided by the micro to indicate the stop will be cleared too.

Figure 12-1 shows a diagram with the main DMA control signals and DMA times.

Figure 12-1. DMA control signals



- **CHANNEL_REQUEST:** One-cycle request signal from the current DMA channel. This signal can be generated simultaneously by several channels but the memory access will be assigned to the most prioritary one.
- **HARD_IDLE_REQUEST:** Request signal for a hard-idle state of the micro. It is a request to stop current micro execution to get access to the external memory, and it must be activated during the complete access.
- **MICRO_STOP:** Signal to indicate that the microcontroller is stopped. It is required by DMA to know if it is possible to access to the memory.

Table 12-1. DMA timing values

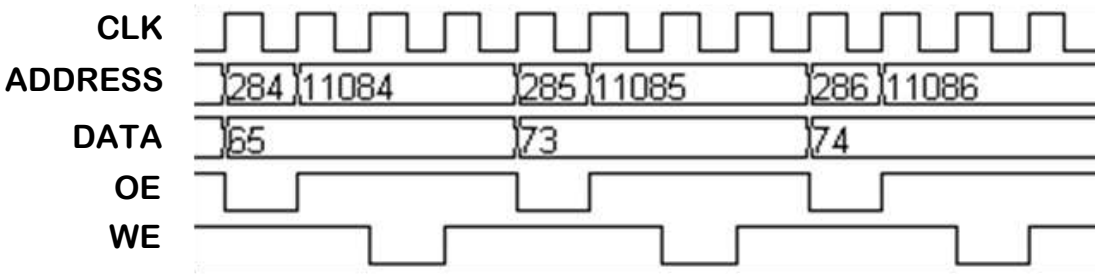
Time	Number of clock cycles
Trequest	1 - 18
Taccess	4*Nbytes
Trestart	1 - 9

- **Trequest:** Time from hard idle request to micro stop. After a channel request, the hard idle request signal has to keep activated till the end of the DMA access. DMA will get access to memory when the stop signal from micro is also activated. This time is variable from one to six machine cycles, depending on instruction execution.
- **Taccess:** Effective access time. This is the total time available for DMA to get full access to the external memory. When access ends, the hard idle request signal has to be deactivated, in order to restart micro execution.
- **Trestart:** Time of micro restart, from the deactivation of the hard idle request to the reactivation of micro execution.

When in DMA mode, only one clock cycle is used to read a byte from memory, and three clock cycles are used to write a byte; thus, a byte-transference consists of 4 clock cycles.

An example is shown in Figure 12-2.

Figure 12-2. Typical DMA read/write process



12.3 DMA Channel Priority

Simultaneous DMA channels activation is possible, so it is needed to establish a priority for each DMA channel. It is possible for two or more channels to request memory access at the same time (low probability), so the channel with the highest priority will get the requested access. Because of hardware requirements, the priority of the channels has been fixed to optimize the physical layer operation. These are the priorities assigned:

Table 12-2. DMA channels priority

DMA Channel	Priority
PHY_RX	3
PHY_TX	2
GEN_1	1
GEN_2	0

Channel priority increases with “priority number”.

DMA channel used in physical reception (PHY_RX) is the most critical, so it has the highest priority, followed by physical transmission DMA channel.

Generic DMA channels (GEN_1 and GEN_2) are identical, except by their priority, and both have lower priority than physical ones.

12.4 DMA Transfer Capabilities

The maximum amount of data which a DMA channel is able to move is 64 KB, corresponding to external data memory size of the ADD8051C3A.

The maximum data size moved in a single DMA access is 256 bytes, and the minimum is 1 byte.

When a DMA channel is active, it will access so many times as necessary to complete the transference, requesting accesses and microcontroller stops.

Transfer sizes are distributed in packets of 16 bytes, in accordance with AES128 data size (see PRIME specification). This is the reference size, and all DMA accesses are configured according to this 16-byte pattern.

Consequently all time accesses are established as well, owing to byte-transference duration (4 clock cycles). “Time access” or “time between accesses” are then referred as a number of these packets. The duration for this packet is 16 bytes * 4 clk/byte * 50 ns/clk = 3.2 µs.

If the DMA access requirement is less than 16 bytes, then access ends when transfer is complete.

For physical DMA channels these times are established by real-time memory transfer requirements. Every physical DMA access for these channels is 16-byte long. Time between consecutive accesses is set up by symbol data of the physical layer when needed.

For generic DMA channels all times are fixed by specific registers. It is needed to consider for each DMA access an additional time to the total time access, which may be variable from 1 clock cycle to 18 clock cycles, depending on ADD8051C3A instruction execution (this variable time differs if the microcontroller is returning from an interruption subroutine or it is returning from Idle mode or Power Down mode).

12.5 DMA Interrupts

In order to inform the ADD8051C3A microcontroller about the completion of a DMA transference, two interrupts are generated by hardware when the transference or a part of it is completed. One of these interrupts is related to the physical layer, and it is activated in both emission and reception. The other interrupt is related to generic channels, and the activation is controlled by specific registers.

Table 12-3. DMA interrupts

Interrupt type	Internal Bit	External Interrupt (ADD8051C3A port)
Physical	INT_PHY PHY_SFR(0)	INT1 (P33)
Generic	INT_DMA PHY_SFR(1)	INT0 (P32)

See PHY_SFR Register in DMA Configuration Registers section.

The physical interruption field is the PHY_SFR(0) bit, named INT_PHY, and it is internally connected to the external interrupt INT1 of the ADD8051C3A microcontroller, which corresponds with P33 port.

The generic channel interruption is also located at the PHY_SFR register.

This interruption field is the PHY_SFR(1) bit, named INT_DMA, and it is internally connected to the external interrupt INT0 of the ADD8051C3A microcontroller, which corresponds to P32 port.

Both interrupts are “low-level active” and are activated by level. When complete or partial transference is finished then the bit will be set to ‘0’ by hardware, generating the corresponding interruption. It the interrupt is active (bit EA –IE(7)– is set to ‘1’ for all interrupts, and bit EX0 –IE(0)– is set to ‘1’ for external interrupt 0 and bit EX1 –IE(2)– is set to ‘1’ for external interrupt 1, see ADD8051 Specification Interrupt Enabling), then this bit must be cleared by writing ‘1’ at the PHY_SFR bit in the interrupt subroutine.

12.6 DMA Flags

There are several flags activated simultaneously with DMA interruptions, that allow to know the type of DMA channel used in the transference, and if this transference is partial or complete. These flags are located at the DMA_SFR register, mapped at address 0xFE2B of the data memory.

See DMA_SFR Register in DMA Configuration Registers section

12.7 Generic DMA channels

Generic DMA channels are fully configurable with specific registers. Physical DMA channels can be understood as a particular case of generic channels, with some of their parameters fixed by physical requirements. There are 7 configurable parameters and 6 registers for each generic channel. Both generic channels are identical except in:

- The priority, which is higher for the generic 1.
- The flags activated at interruption.

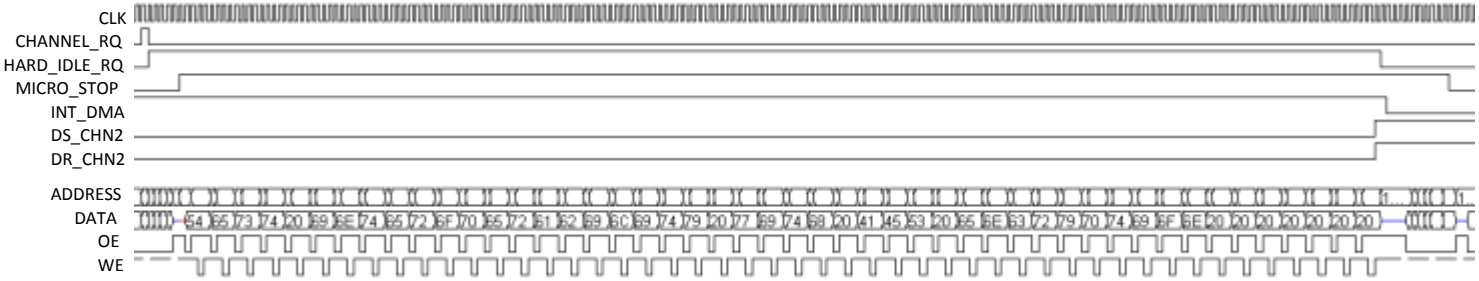
The priority for both channels is lower than priority for physical channels.

The registers involved in DMA channels operation are described in Peripheral registers section.

ADDR_DMA1_SRC	ADDR_DMA2_SRC
ADDR_DMA1_DTN	ADDR_DMA2_DTN
SIZE_DMA1_SRC	SIZE_DMA2_SRC
SIZE_DMA1_DTN	SIZE_DMA2_DTN
SIZE_RQ_DMA1	SIZE_RQ_DMA2
DMA1_CHN_CTL	DMA2_CHN_CTL

As an example, below it is shown a memory-transference with channel 2, by establishing source and destination sizes equal to 48 bytes, the same as the request size, the number of packets for DMA access and between accesses equal to 15. It is possible to see the activation of DS_CHN2, DR_CHN2 and the interruption INT_DMA and memory control signals.

Figure 12-3. DMA channel 2 process



12.8 Physical DMA channels

Physical DMA channels can be understood as a particular case of generic channels in which some parameters are fixed. The main differences between PHY_RX (Reception physical DMA channel) and PHY_TX (Transmission physical DMA channel) channels are the direction of the transference, the interrupts generated, and the flags activated. Note that these DMA channels operate against internal registers mapped in data memory, so one of the buffers is always internal.

12.9 PHY_RX DMA channel

This is a dedicated DMA channel for physical data reception. The source is a 16-byte width buffer at the end of the reception branch (see PHY layer section), and the destination buffer is the data memory.

Size of the destination buffer is variable and corresponds to PPDU size (DMA destination size = PPDU length). Time between accesses depends on symbol duration, modulation schemes and physical layer processing capability.

There are two interrupts generated at reception, one of them when the physical header is correctly received (two first symbols), and the other one when the message is completely received. Then it is easy to see that DMA request size in this case correspond to the physical header size (11 bytes), which causes an interruption when the current transference size is equal to PHY header size and also activates an internal flag. For the other interruption another internal flag is activated to indicate the completion of reception.

The priority for this channel is the maximum available, because of critical reception requirements: physical data must be moved as fast as possible after symbol processing.

12.10 PHY_TX DMA channel

This is a dedicated DMA channel for physical data transmission. The source buffer is data memory and the destination is a 16-byte width buffer at the beginning of the transmission branch.

Size of the source buffer is also variable and corresponds to PPDU size (DMA source size = PPDU length). Time between accesses NPUM also depends on physical requirements.

For this channel there is only one interrupt available, at the end of the transmission, so it is easy to see that DMA request size in this case is equal to DMA source size. Note that this is the only DMA channel in which the end of memory transference is caused by the completion of the source buffer size, instead of the rest of the channels, in which the end of DMA is provoked by the destination buffer size. At the end of transmission, a physical interruption is generated and an internal flag is set.

The priority for this channel is lower than PHY_RX channel, but higher than generic channels, due to transmission requirements: physical data must be prepared before symbol processing.

12.11 DMA Configuration Registers

12.11.1 PHY_SFR Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
PHY_SFR	BCH_ERR	CD	UMD	--	--	TXRX	INT_DMA	INT_PHY

Name: PHY_SFR

Address: 0xFE2A

Access: Read/write

Reset: 0x87

- **--:** Reserved bit
- **BCH_ERR:** Busy Channel Error Flag.

  This bit is cleared to ‘0’ by hardware to indicate the presence of an OFDM signal at the transmission instant. Otherwise, this field value is ‘1’.

  This bit is used for returning a result of “Busy Channel” in the PHY_DATA.confirm primitive (see PRIME specification).
- **CD:** Carrier Detect bit.

  This bit is set to ‘1’ by hardware when an OFDM signal is detected, and it is active during the whole reception.

  This bit is used in channel access (CSMA-CA algorithm) for performing channel-sensing.
- **UMD:** Unsupported Modulation Scheme flag.

  This flag is set to ‘1’ by hardware every time a header with correct CRC is received, but the PROTOCOL field in this header indicates a modulation scheme not supported by the system.
- **TXRX:** Transmission order.

  When data to transmit is ready at ADDR_PHY_INI_TX in data memory, the Time value is set at TX_TIME register and then the emission level is specified at ATTENUATION register, then TXRX bit has to be set to '0' in order to init transmission.

  If this bit is read, only returns '0' when physical transmission has started. Otherwise, it returns '1'.

  The transmission will begin when TIMER_BEACON_REF is equal to TX_TIME.
- **INT_DMA:** DMA interruption

  This bit is internally connected to the ADD8051C3A microcontroller external interrupt 0 (INT0), which corresponds with P32 port.

  It is low-level active. It is set to '0' by DMA hardware and is cleared by writing '1' in the bit PHY_SFR(1).
- **INT_PHY:** Physical Layer interruption

  This bit is internally connected to the ADD8051C3A microcontroller external interrupt 1 (INT1), which corresponds with P33 port.

  It is low-level active. It is set to '0' by physical layer and is cleared by writing '1' in the bit PHY_SFR(0).

12.11.2 DMA_SFR Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
DMA_SFR	DR_CHN2	DS_CHN2	DR_CHN1	DS_CHN1	DR_TX	DS_TX	DR_RX	DS_RX

Name: DMA_SFR

Address: 0xFE2B

Access: Read/write

Reset: 0x04

- **DR_CHN2:** Data Request flag for generic DMA channel 2
  This flag is activated when the requested size of the destination buffer is already transferred.
  The requested size is specified at SIZE_RQ_DMA2.
- **DS_CHN2:** Data Size flag for generic DMA channel 2
  This flag is activated when the complete size of the destination buffer is already transferred.
  The total transference size is specified at SIZE_DMA2_DTN.
  The source buffer is data memory at ADDR_DMA2_SRC.
  The destination buffer is data memory at ADDR_DMA2_DTN.
- **DR_CHN1:** Data Request flag for generic DMA channel 1
  This flag is activated when the requested size of the destination buffer is already transferred.
  The requested size is specified at SIZE_RQ_DMA1.
- **DS_CHN1:** Data Size flag for generic DMA channel 1
  This flag is activated when the complete size of the destination buffer is already transferred.
  The total transference size is specified at SIZE_DMA1_DTN.
  The source buffer is data memory at ADDR_DMA1_SRC.
  The destination buffer is data memory at ADDR_DMA1_DTN.
- **DR_TX:** Data Request flag for PHY_TX DMA channel
  This bit is always'0' because the PHY_TX buffer is only interrupted at the end of the transmission. (In reception there are two interruptions, the first one when the header is received and the other one when the complete message is receive. In transmission there is no interruption when the header is sent)
- **DS_TX:** Data Size flag for PHY_TX DMA channel
  This flag is activated when the complete size of the source buffer is transferred.
  That means that a complete message has been correctly transmitted.
  The source buffer is data memory at ADDR_PHY_INI_TX.
  The destination buffer is PHY_TX buffer.
  The transference size is the complete PPDU(PHY Protocol Data Unit) length.
- **DR_RX:** Data Request flag for PHY_RX DMA channel
  This flag is activated when the requested size of the destination buffer is already transferred.
  That means that an incoming message physical header has been correctly received.
  The requested size is fixed to 11 bytes, in accordance with the physical layer header.

- **DS_RX:** Data Size flag for PHY_RX DMA channel

  This flag is activated when the complete size of the destination buffer is transferred.

  That means that a complete message has been correctly received.

  The source buffer is PHY_RX and destination buffer is data memory at ADDR_PHY_INI_RX.

  The transference size is the complete PPDU (PHY protocol Data Unit) length.

12.11.3 PHY_TX Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
PHY_TX	PHY_TX(127:120)								@FE00
	…							…	
	PHY_TX(7: 0)								@FE0F

Name: PHY_TX

Address: 0xFE00 – 0xFE0F

Access: Read only

Reset: 0x00, …, 0x00;

- **PHY_TX:** Physical layer input buffer for transmission. The buffer size is 16 bytes in order to fit in with standard AES128 data size.

  Specific transfer from SRAM to this buffer is executed by DMA when necessary in transmission. DMA TX channel is dedicated for this type of transference.

DMA source address	=>	ADDR_PHY_INI_TX
DMA destination address	=>	PHY_TX(0)
DMA source size	=>	variable
DMA destination size	=>	16

12.11.4 PHY_RX Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
PHY_RX	PHY_RX(127:120)								@FE10
	…							…	
	PHY_RX(7: 0)								@FE1F

Name: PHY_RX

Address: 0xFE10 – 0xFE1F

Access: Read only

Reset: 0x00, …, 0x00

- **PHY_RX:** Physical layer output buffer for reception. The buffer size is 16 bytes in order to fit in with standard AES128 data size.

  Specific transfer from SRAM to this buffer is executed by DMA when necessary in reception. DMA RX channel is dedicated for this type of transference.

DMA source address	=>	PHY_RX(0)
DMA destination address	=>	ADDR_PHY_INI_RX
DMA source size	=>	16
DMA destination size	=>	variable

12.11.5 ADDR_PHY_INI_RX Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
ADDR_PHY_INI_RX	ADDR_PHY_INI_RX(15:8)								@0xFE20
	ADDR_PHY_INI_RX(7:0)								@0xFE21

Name: ADDR_PHY_INI_RX

Address: 0xFE20 – 0xFE21

Access: Read/write

Reset: 0x02, 0x10

- **ADDR_PHY_INI_RX:** Data memory address for reception

  This is the destination address to which DMA RX channel initiates data transfer from the physical layer

12.11.6 ADDR_PHY_INI_TX Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
ADDR_PHY_INI_TX	ADDR_PHY_INI_TX(15:8)								@0xFE22
	ADDR_PHY_INI_TX(7:0)								@0xFE23

Name: ADDR_PHY_INI_TX

Address: 0xFE22 – 0xFE23

Access: Read/write

Reset: 0x00, 0x10

- **ADDR_PHY_INI_TX:** Data memory address for transmission

  This is the source address from which DMA TX channel initiates data transfer to the physical layer

12.11.7 ADDR_DMA1_SRC Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
ADDR_DMA1_SRC	ADDR_DMA1_SRC(15:8)								@0xFEE0
	ADDR_DMA1_SRC (7:0)								@0xFEE1

Name: ADDR_DMA1_SRC

Address: 0xFEE0 – 0xFEE1

Access: Read/write

Reset: 0x03, 0x00

- **ADDR_DMA1_SRC:** The 16-bit value stored in these registers is the source address for DMA channel 1.

  Start address at data memory from which source data is copied to the destination buffer. DMA begins reading from this address and increments it one by one, after each byte reading

12.11.8 ADDR_DMA1_DTN Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
ADDR_DMA1_DTN	ADDR_DMA1_DTN(15:8)								@0xFEE2
	ADDR_DMA1_DTN (7:0)								@0xFEE3

Name: ADDR_DMA1_DTN

Address: 0xFEE2 – 0xFEE3

Access: Read/write

Reset: 0x05, 0x00

- **ADDR_DMA1_DTN:** The 16-bit value stored in these registers is the destination address for DMA channel 1.

  Start address at data memory to which data is copied from the source buffer. DMA begins writing to this address and increments it one by one, after each byte-writing.

12.11.9 SIZE_DMA1_SRC Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
SIZE_DMA1_SRC	SIZE_DMA1_SRC(15:8)								@0xFEE4
	SIZE_DMA1_SRC (7:0)								@0xFEE5

Name: SIZE_DMA1_SRC

Address: 0xFEE4 – 0xFEE5

Access: Read/write

Reset: 0x00, 0x10

- **SIZE_DMA1_SRC:** The 16-bit value stored in these registers is the source size for DMA channel 1

  Note that source buffer size and destination buffer size can be different:

  - If source buffer size is smaller than destination buffer size, then DMA will restart the source address at ADDR_DMA1_SRC when source buffer size is reached, in order to full fill the destination buffer with data from the source buffer.
  - If source buffer size is greater than destination buffer size, then only SIZE_DMA1_DTN bytes will be copied and the rest will be ignored.

  Nevertheless, in normal operation both source and destination buffers have the same size.

12.11.10 SIZE_DMA1_DTN Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
SIZE_DMA1_DTN	SIZE_DMA1_DTN(15:8)								@0xFEE6
	SIZE_DMA1_DTN (7:0)								@0xFEE7

Name: SIZE_DMA1_DTN

Address: 0xFEE6 – 0xFEE7

Access: Read/write

Reset: 0x01, 0x00

- **SIZE_DMA1_DTN:** The 16-bit value stored in these registers is the destination size for DMA channel 1

  This is the required size to complete the DMA transference. The channel will be active until this size is transferred.

  The completion of this transference causes an interrupt by clearing to '0' the INT_DMA bit (PHY_SFR(1)), and the activation of the DS_CHN1 flag to '1' (when using GEN_1 channel) or DS_CHN2 (when using GEN_2 channel)

12.11.11 SIZE_RQ_DMA1 Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
SIZE_RQ_DMA1	SIZE_RQ_DMA1(15:8)								@0xFEE8
	SIZE_RQ_DMA1(7:0)								@0xFEE9

Name: SIZE_RQ_DMA1

Address: 0xFEE8 – 0xFEE9

Access: Read/write

Reset: 0x01, 0x00

- **SIZE_RQ_DMA1:** The 16-bit value stored in these registers is the request size for DMA channel 1

  This size has to be smaller or equal than SIZE_DMA1_DTN, and it is used to detect a partial completion of the transference. The completion of this size causes an interrupt by clearing to '0' the INT_DMA bit (PHY_SFR(1)) and the activation of DR_CHN1 flag to '1' (when using GEN_1) or DR_CHN2 (when using GEN_2).

  Note that one memory transfer can cause two interrupts at different time. If SIZE_RQ_DMA1 is equal to SIZE_DMA1_DTN, then only one interrupt will be generated and both DS_CHN1 and DR_CHN1 flags will be activated when using GEN_1 (DS_CHN2 and DR_CHN2 when using GEN_2)

12.11.12 DMA1_CHN_CTL Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
DMA1_CHN_CTL	NPDMA(3:0)				NPUM(3:0)			

Name: DMA1_CHN_CTL

Address: 0xFEEA

Access: Read/write

Reset: 0x00

This register is used in DMA channel 1 control. It is possible to configure the time of each access or the time between consecutive accesses. These times are referred to 16-bytes packets, with a duration of 3.2µs.

- **NPDMA:** Number of Packets for DMA

  Time (in 3.2µs steps) of every DMA access when the DMA channel 1 is active. Nevertheless, the real time spent depends on the microcontroller execution.

  When DMA channel 1 is not active, this value is cleared to “0”. Modifying this value causes the activation of the channel. If NPDMA>0 then the channel will be activated. When the channel completes the transference, NPDMA value is cleared by hardware (NPDMA=0)

- **NPUM:** Number of Packets for UM (microcontroller).

  This field is used to set the time (in 3.2µs steps) between DMA accesses when the DMA channel 1 is active.

  This time cannot be null, so the number of packets used is NPUM+1.

12.11.13 ADDR_DMA2_SRC Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
ADDR_DMA2_SRC	ADDR_DMA2_SRC(15:8)								@0xFEEB
	ADDR_DMA2_SRC (7:0)								@0xFEEC

Name: ADDR_DMA2_SRC

Address: 0xFEEB – 0xFEEC

Access: Read/write

Reset: 0x07, 0x00

- **ADDR_DMA2_SRC:** The 16-bit value stored in these registers is the source address for DMA channel 2. Start address at data memory from which source data is copied to the destination buffer. DMA begins reading from this address and increments it one by one, after each byte reading

12.11.14 ADDR_DMA2_DTN Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
ADDR_DMA2_DTN	ADDR_DMA2_DTN(15:8)								@0xFEED
	ADDR_DMA2_DTN (7:0)								@0xFEEE

Name: ADDR_DMA2_DTN

Address: 0xFEED – 0xFEEE

Access: Read/write

Reset: 0x09, 0x00

- **ADDR_DMA2_DTN:** The 16-bit value stored in these registers is the destination address for DMA channel 2.

  Start address at data memory to which data is copied from the source buffer. DMA begins writing to this address and increments it one by one, after each byte-writing.

12.11.15 SIZE_DMA2_SRC Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
SIZE_DMA2_SRC	SIZE_DMA2_SRC(15:8)								@0xFEEF
	SIZE_DMA2_SRC (7:0)								@0xFEF0

Name: SIZE_DMA2_SRC

Address: 0xFEEF – 0xFEF0

Access: Read/write

Reset: 0x00, 0x10

- **SIZE_DMA2_SRC:** The 16-bit value stored in these registers is the source size for DMA channel 2

  Note that source buffer size and destination buffer size can be different:

  - If source buffer size is smaller than destination buffer size, then DMA will restart the source address at ADDR_DMA2_SRC when source buffer size is reached, in order to full fill the destination buffer with data from the source buffer.
  - If source buffer size is greater than destination buffer size, then only SIZE_DMA2_DTN bytes will be copied and the rest will be ignored.

  Nevertheless, in normal operation both source and destination buffers have the same size.

12.11.16 SIZE_DMA2_DTN Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
SIZE_DMA2_DTN	SIZE_DMA2_DTN(15:8)								@0xFEF1
	SIZE_DMA2_DTN (7:0)								@0xFEF2

Name: SIZE_DMA2_DTN

Address: 0xFEF1 – 0xFEF2

Access: Read/write

Reset: 0x01, 0x00

- **SIZE_DMA2_DTN:** The 16-bit value stored in these registers is the destination size for DMA channel 2

  This is the required size to complete the DMA transference. The channel will be active until this size is transferred.

  The completion of this transference causes an interrupt by clearing to '0' the INT_DMA bit (PHY_SFR(1)), and the activation of the DS_CHN1 flag to '1' (when using GEN_1 channel) or DS_CHN2 (when using GEN_2 channel)

12.11.17 SIZE_RQ_DMA2 Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
SIZE_RQ_DMA2	SIZE_RQ_DMA2(15:8)								@0xFEF3
	SIZE_RQ_DMA2(7:0)								@0xFEF4

Name: SIZE_RQ_DMA2

Address: 0xFEF3 – 0xFEF4

Access: Read/write

Reset: 0x01, 0x00

- **SIZE_RQ_DMA2:** The 16-bit value stored in these registers is the request size for DMA channel 2

  This size has to be smaller or equal than SIZE_DMA2_DTN, and it is used to detect a partial completion of the transference. The completion of this size causes an interrupt by clearing to '0' the INT_DMA bit (PHY_SFR(1)) and the activation of DR_CHN1 flag to '1' (when using GEN_1) or DR_CHN2 (when using GEN_2).

  Note that one memory transfer can cause two interrupts at different time. If SIZE_RQ_DMA2 is equal to SIZE_DMA1_DTN, then only one interrupt will be generated and both DS_CHN1 and DR_CHN1 flags will be activated when using GEN_1 (DS_CHN2 and DR_CHN2 when using GEN_2)

12.11.18 DMA2_CHN_CTL Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
DMA2_CHN_CTL	NPDMA(3:0)				NPUM(3:0)			

Name: DMA2_CHN_CTL

Address: 0xFEF5

Access: Read/write

Reset: 0x00

This register is used in DMA channel 2 control. It is possible to configure the time of each access or the time between consecutive accesses. These times are referred to 16-bytes packets, with a duration of 3.2µs.

- **NPDMA:** Number of Packets for DMA

  Time (in 3.2µs steps) of every DMA access when the DMA channel 2 is active. Nevertheless, the real time spent depends on the microcontroller execution.

  When DMA channel 2 is not active, this value is cleared to “0”. Modifying this value causes the activation of the channel. If NPDMA>0 then the channel will be activated. When the channel completes the transference, NPDMA value is cleared by hardware (NPDMA=0)

- **NPUM:** Number of Packets for UM (microcontroller).

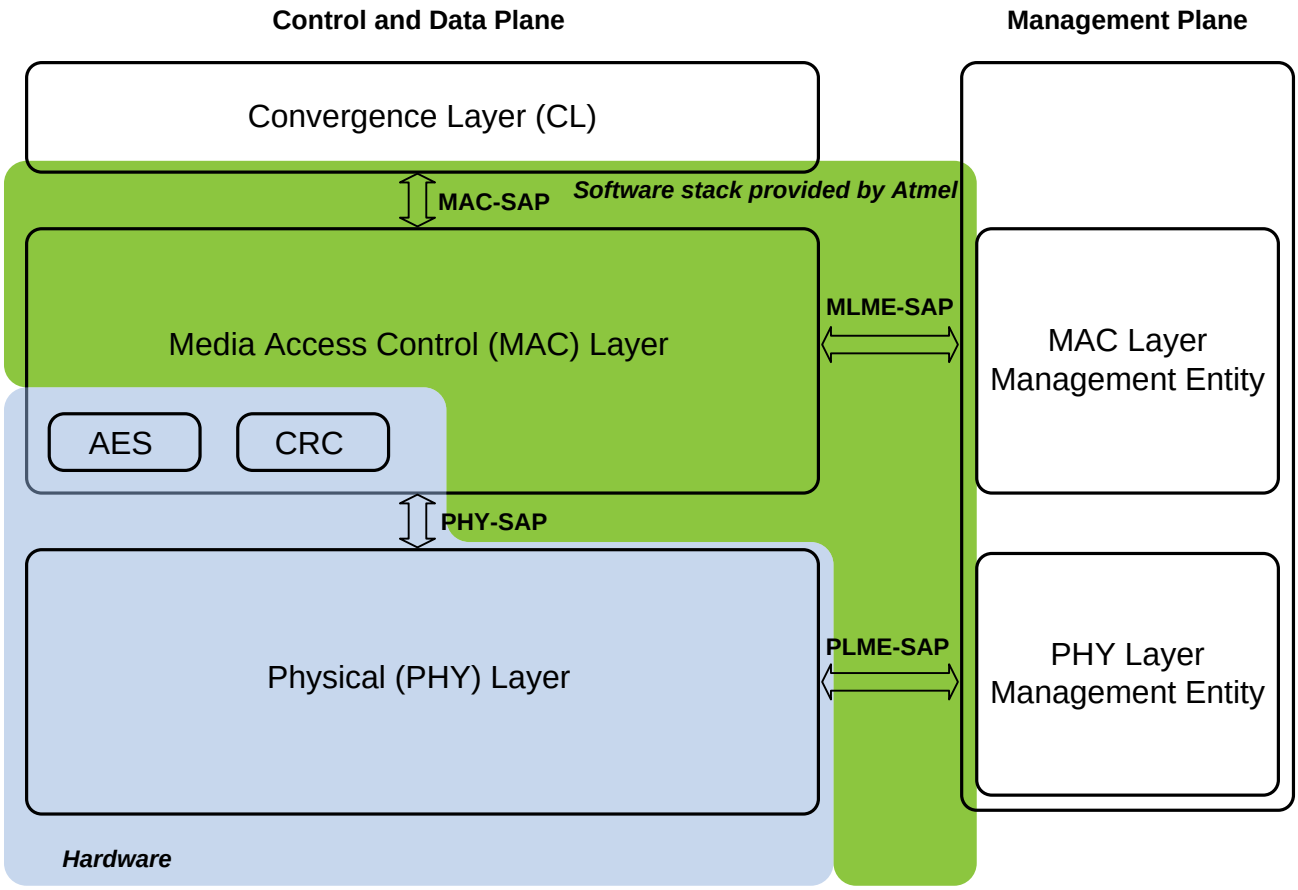
  This field is used to set the time (in 3.2µs steps) between DMA accesses when the DMA channel 2 is active.

  This time cannot be null, so the number of packets used is NPUM+1.

13. ATPL210 MAC Coprocessor

The ATPL210 hardware MAC layer consists of a hardware implementation of some functionalities of the MAC Layer Entity specified in PRIME specification. These features are CRC calculation and AES128 block. So it is possible to consider this hardware as a MAC coprocessor to improve the ADD8051C3A performance.

Figure 13-1. ATPL210 Software Stack Diagram



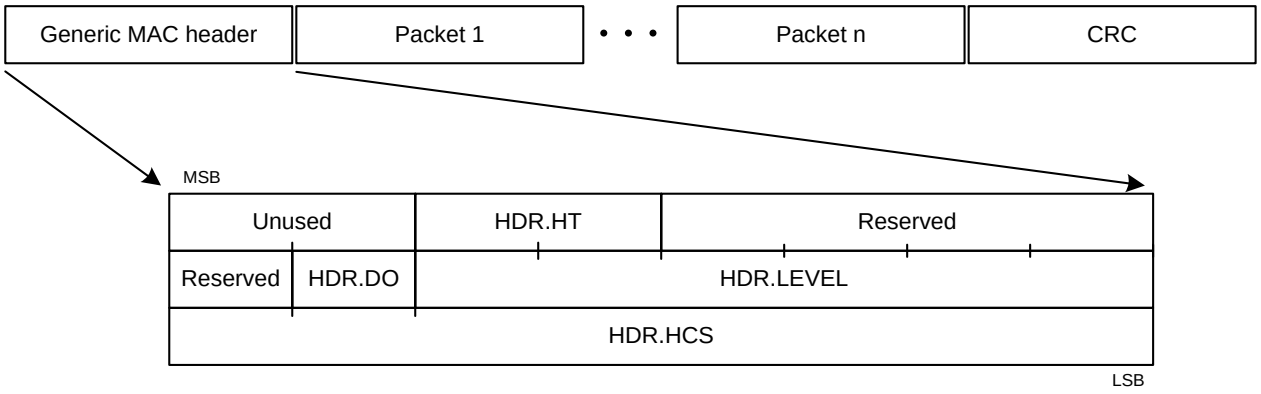
Atmel PRIME stack implements by software the rest of the MAC layer requirements and capabilities. Furthermore, the software package allows the communication with the Management Plane by means of the two Access points described by PRIME (PHY Layer Management Entity SAP and MAC Layer Management Entity SAP) and the interface to communicate MAC layer with the upper layer (Convergence Layer).

Please check the “Atmel PRIME Stack User Manual” for software package detailed description and functionality.

13.2 Cyclic Redundancy Check (CRC)

There are three types of MAC PDUs (generic, promotion and beacon) for different purposes, and each one has its own specific CRC. In ATPL210A there is a hardware implementation of every CRC type calculated by the MAC layer. This CRC hardware-calculation is enabled by default. Note that the CRC included at the physical layer is also a hardware implementation available in ATPL210A and it is also enabled by default.

Figure 13-2. Generic MAC PDU format and generic MAC header detail



In transmission all CRC bytes are real-time calculated and the last bytes of the MAC PDU are overwritten with these values, (provided that the field HT in the first byte of the MAC header in transmission data is equal to the corresponding MAC PDU type).

In reception the CRC bytes are also real-time calculated and these bytes are checked with the last bytes of the MAC PDU. If the CRC is not correct, then an error flag is activated, the complete frame is discarded, and the corresponding error counter is increased. These counters allow the MAC layer to take decisions according to error ratio.

For the Generic MAC PDU, there is an 8-bit CRC in the Generic MAC header, which corresponds to PRIME HDR.HCS. In reception if this CRC doesn't check successfully, the current frame is discarded and no interruption is generated.

This works in the same way as CRC for the PHY layer (CRC Ctrl, located in the PHY header, see PRIME specification for further information).

There is another CRC for the Generic MAC PDU which is the last field of the GPDU. It is 32 bits long and it is used to detect transmission errors. The CRC shall cover the concatenation of the SNA with the GPDU except for the CRC field itself. In reception, if the CRC is not successful then an internal flag is set and the error counter is increased.

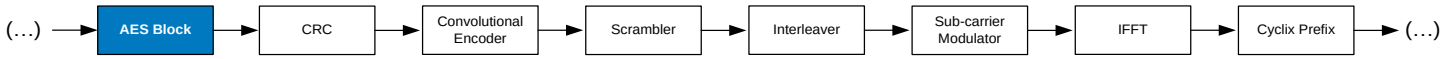
For the Promotion Needed PDU there is an 8-bit CRC, calculated with the first 13 bytes of the header. In reception, if this CRC is not correct, then an internal flag is set and the corresponding error counter is increased.

For the Beacon PDU there is a 32-bit CRC calculated with the same algorithm as the one defined for the CRC of the Generic MAC PDU. This CRC shall be calculated over the complete BPDU except for the CRC field itself. In reception, if this CRC is not successful, then an internal flag is set and the same error counter as for GPDU is increased. The hardware used for this CRC is the same as the one used for GPDU.

13.3 Advanced Encryption Standard

One of the security functionalities in PRIME is the 128-bit AES encryption of data and its associated CRC. ATPL210A includes a hardware implementation of this block, and it is used by the physical layer in real-time transmission/reception. It is possible to use this block externally as a peripheral unit, by accessing the specific registers designed to control it. Therefore there are some configurable parameters and input/output buffers to the block.

Figure 13-3. PHY Layer transmitter block diagram



There are two basic operation ways in ATPL210A when using PRIME Security Profile 1. The first one is real-time encryption and the second one is independent encryption from the PHY layer.

Real-Time Encryption: the AES128 core is integrated in the physical chain, and data is encrypted and decrypted in real-time when needed. In transmission, data is transferred to the emission buffer by means of the DMA TX channel. Then the 128 bits located in the buffer are encrypted before starting transmission (Note that Beacon PDU, Promotion PDU and Generic MAC header, as well as several control packets, are not encrypted). Data is extracted when required from this buffer until it is empty, and then a new DMA transfer is requested to fill the 16 bytes and a new encryption is executed. The key used for encryption must be set at the corresponding register, and it can vary from a packet to another.

In reception, data is obtained from the PHY layer and it is passed to the AES128 block. When the reception buffer is full with incoming data, the 128 bits are decrypted and transferred to external memory through DMA RX channel. Then the reception buffer is available again to fill with processed data.

The header is always real-time analyzed in order to know if encryption process must be applied.

Independent Encryption: the AES128 core is used as a peripheral unit, accessible with several registers mapped in external memory. In this mode, when in transmission, data must be encrypted previously to the use of the PHY_DATA.request primitive (see PRIME specification), in an independent way. In reception, data passed by the PHY layer is already encrypted and must be decrypted in a subsequent process.

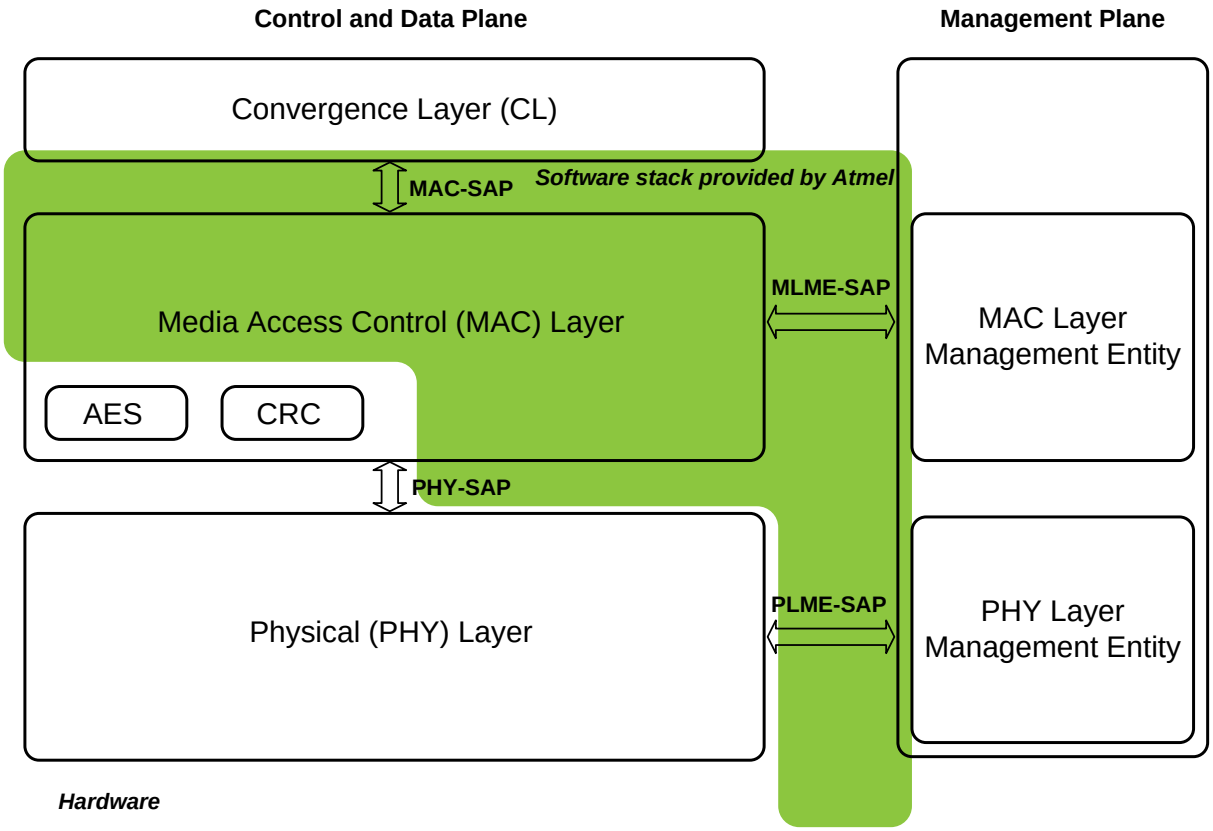
When working with AES block as a peripheral unit, automatic CRC calculation by hardware is disabled.

13.4 Atmel PRIME Software Stack

Looking for the best compromise between efficiency and versatility, Atmel has developed a PRIME software stack that implements and fulfills the capabilities of the MAC layer not carried out by ATPL210 hardware.

Atmel PRIME stack allows the final user to control the system by means of four sets of primitives, making the low level functions and the PHY-MAC interface transparent to the user.

Figure 13-4. Atmel PRIME Software Stack diagram

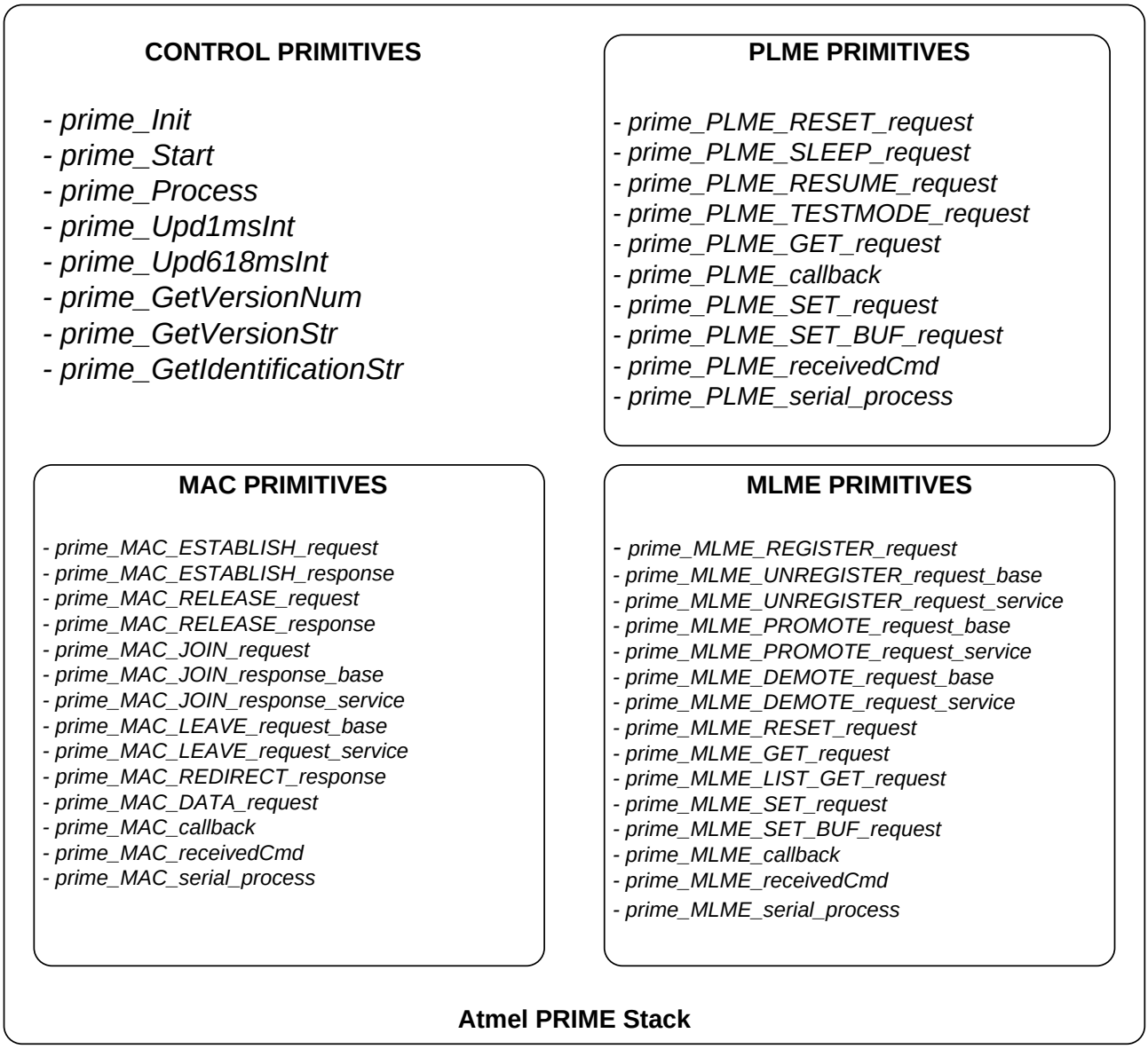


By using Atmel PRIME stack, ATPL210 PHY and MAC layers can be controlled by means of the three access points shown in the figure: MAC-SAP (MAC Service Access Point), MLME-SAP (MAC Layer Management Entity Service Access Point) and PLME-SAP (PHY Layer Management Entity Service Access point). Furthermore, there are general functions to manage the software package.

An example of high level functions that manage the software stack is shown in Figure 13-5.

Please check the “Atmel PRIME Stack User Manual” for software package detailed and updated description and functionality.

Figure 13-5. Atmel PRIME version 1.3.04.04 stack functions



13.5 MAC Coprocessor Registers

13.5.1 SNA Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
SNA	SNA(47:40)								@FE62
	…							…	
	SNA(7: 0)								@FE67

Name: SNA

Address: 0xFE62 – 0xFE67

Access: Read/write

Reset: 0x00, …, 0x00

- **SNA:** Sub Network Address

  These registers store the 48-bit Sub Network Address. When the system Sub Network Address is available, the microcontroller must write it down so the Physical layer will be able to correctly calculate the CRC's, which depend on this parameter

13.5.2 VITERBI_BER_HARD Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
VITERBI_BER_HARD	VITERBI_BER_HARD(7:0)							

Name: VITERBI_BER_HARD

Address: 0xFE36

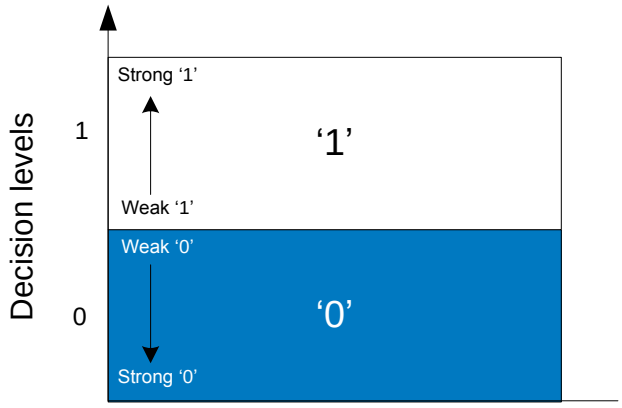
Access: Read only

Reset: 0x00

- **VITERBI_BER_HARD:** This register stores the number of errors accumulated in a message reception using Viterbi hard* decision. The value is cleared by hardware each time a new message is received.

**Hard Decision: in "hard" detection there are only two decision levels. If the received value is different than the corrected one, the error value taken is "1". Otherwise, the error value taken is "0".*

Figure 13-6. Viterbi Hard detection decision levels



From the value in VITERBI_BER_HARD register it is possible to calculate de Bit Error Rate according to the following formula:

13.5.3 VITERBI_BER_SOFT Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
VITERBI_BER_SOFT	VITERBI_BER_SOFT(7:0)							

Name: VITERBI_BER_SOFT

Address: 0xFE37

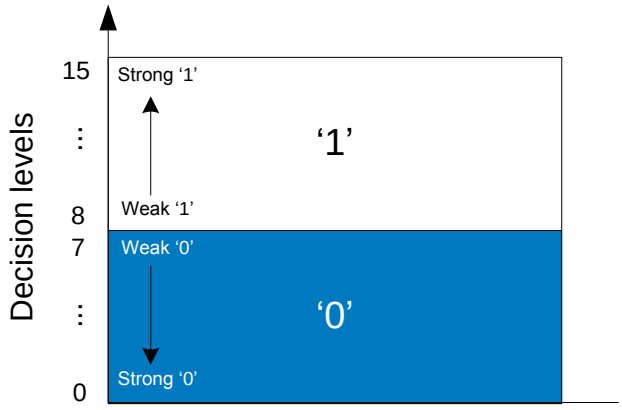
Access: Read only

Reset: 0x00

- **VITERBI_BER_SOFT:** This register stores a value proportional to the number of errors accumulated in a message reception using Viterbi soft* decision. The value is cleared by hardware each time a new message is received.

**Soft Decision: in "soft" decision there are fifteen decision levels. A strong '0' is represented by a value of "0", while a strong '1' is represented by a value of "15". The rest of values are intermediate, so "7" is used to represent a weak '0' and "8" represents a weak '1'. Soft decision calculates the error in one bit received as the distance in decision levels between the value received (a value in the range 0 to 15) and the corrected one (0 or 15).*

Figure 13-7. Viterbi Hard detection decision levels



13.5.4 ERR_CRC32_MAC Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
ERR_CRC32_MAC	ERR_CRC32_MAC(15:8)								@0xFEBA
	ERR_CRC32_MAC(7:0)								@0xFEBB

Name: ERR_CRC32_MAC

Address: 0xFEBA – 0xFEBB

Access: Read/write

Reset: 0x00, 0x00

- **ERR_CRC32_MAC:** 16-bit value that stores the number of received messages that have been discarded by an error in the MAC layer CRC32.
  Note: to clear this value, these registers must be reset by the microcontroller.

13.5.5 ERR_CRC8_MAC Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
ERR_CRC8_MAC	ERR_CRC8_MAC(15:8)								@0xFEBC
	ERR_CRC8_MAC(7:0)								@0xFEBD

Name: ERR_CRC8_MAC

Address: 0xFEBC – 0xFEBD

Access: Read/write

Reset: 0x00, 0x00

- **ERR_CRC8_MAC:** 16-bit value that stores the number of received messages that have been discarded by an error in the payload MAC layer CRC8.
  Note: to clear this value, these registers must be reset by the microcontroller.

13.5.6 ERR_CRC8_AES Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
ERR_CRC8_AES	ERR_CRC8_AES(15:8)								@0xFEBE
	ERR_CRC8_AES(7:0)								@0xFEBF

Name: ERR_CRC8_AES

Address: 0xFEBE – 0xFEBF

Access: Read/write

Reset: 0x00, 0x00

- **ERR_CRC8_AES:** 16-bit value that stores the number of received messages that have been discarded by an error in the payload AES CRC8.
  Note: to clear this value, these registers must be reset by the microcontroller.

13.5.7 ERR_CRC8_MAC_HD Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
ERR_CRC8_MAC_HD	ERR_CRC8_MAC_HD(15:8)								@0xFEC0
	ERR_CRC8_MAC_HD(7:0)								@0xFEC1

Name: ERR_CRC8_MAC_HD

Address: 0xFEC0 – 0xFEC1

Access: Read/write

Reset: 0x00, 0x00

- **ERR_CRC8_MAC_HD:** 16-bit value that stores the number of received messages that have been discarded by an error in the header MAC layer.

  Note: to clear this value, these registers must be reset by the microcontroller.

13.5.8 ERR_CRC8_PHY Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
ERR_CRC8_PHY	ERR_CRC8_PHY(15:8)								@0xFEC2
	ERR_CRC8_PHY(7:0)								@0xFEC3

Name: ERR_CRC8_PHY

Address: 0xFEC2 – 0xFEC3

Access: Read/write

Reset: 0x00, 0x00

- **ERR_CRC8_PHY:** 16-bit value that stores the number of received messages that have been discarded by an error in the PHY layer CRC8.
  Note: to clear this value, these registers must be reset by the microcontroller.

13.5.9 FALSE_DET_CONFIG Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
FALSE_DET _CONFIG	--		ERR_CRC8 _MAC	INVALID _PROTOCOL	ERROR _LEN	ERROR_PAD _LEN	UNKNOWN _PDU	UNKNOWN _SP

Name: FALSE_DET_CONFIG

Address: 0xFEC4

Access: Read/write

Reset: 0x10

- **--:** Reserved bits
- **ERR_CRC8_MAC:** If this bit is set to '1', FALSE_DET registers will increase its error counter if a received message has a correct PHY layer CRC8 but the MAC layer CRC8 present in its header is wrong
- **INVALID_PROTOCOL:** If this bit is set to '1', FALSE_DET registers will increase its error counter if a received message has a correct PHY layer CRC8 but the PROTOCOL field indicates a modulation not supported by the system
- **ERROR_LEN:** If this bit is set to '1', FALSE_DET registers will increase its error counter if a received message has a correct PHY layer CRC8 but the LEN field indicates a not valid message length
- **ERROR_PAD_LEN:** If this bit is set to '1', FALSE_DET registers will increase its error counter if a received message has a correct PHY layer CRC8 but the PAD_LEN field indicates a not valid message padding length
- **UNKNOWN_PDU:** If this bit is set to '1', FALSE_DET registers will increase its error counter if a received message has a correct PHY layer CRC8 but the HT field indicates a header type different from BEACON, PROMOTION or GENERIC
- **UNKNOWN_SP:** If this bit is set to '1', FALSE_DET registers will increase its error counter if a received message has a correct PHY layer CRC8 but the SECURITY_PROTOCOL field is wrong.

13.5.10 FALSE_DET Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
FALSE_DET	FALSE_DET(15:8)								@0xFEC5
	FALSE_DET(7:0)								@0xFEC6

Name: FALSE_DET

Address: 0xFEC5 – 0xFEC6

Access: Read/write

Reset: 0x00, 0x00

- **FALSE_DET:** Erroneous non-discarded messages.

  16-bit value that stores the number of received messages that have not been discarded since its PHY layer CRC8 is correct, but in which there are other/others incorrect field/fields. The fields to be taken into account to increase the counter in case they were wrong can be selected by FALSE_DET_CONFIG register.

  Note: to clear this value, these registers must be reset by the microcontroller

13.5.11 MAX_LEN_DBPSK Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
MAX_LEN_DBPSK	--		MAX_LEN_DBPSK(5:0)					

Name: MAX_LEN_DBPSK

Address: 0xFEC8

Access: Read/write

Reset: 0xFF

- **--:** Reserved bits
- **MAX_LEN_DBPSK:** This register sets the maximum length, measured in OFDM symbols, that the system allows to receive when working with DBPSK modulation and no Viterbi encoding.

  If a message in such modulation/encoding is received and its LEN field indicates a length above the threshold defined by MAX_LEN_DBPSK value, the message will be discarded.

13.5.12 MAX_LEN_DBPSK_VTB Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
MAX_LEN_DBPSK_VTB	--		MAX_LEN_DBPSK_VTB(5:0)					

Name: MAX_LEN_DBPSK_VTB

Address: 0xFEC9

Access: Read/write

Reset: 0xFF

- **--:** Reserved bits
- **MAX_LEN_DBPSK_VTB:** This register sets the maximum length, measured in OFDM symbols, that the system allows to receive when working with DBPSK modulation and Viterbi encoding.

  If a message in such modulation/encoding is received and its LEN field indicates a length above the threshold defined by MAX_LEN_DBPSK_VTB value, the message will be discarded.

13.5.13 MAX_LEN_DQPSK Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
MAX_LEN_DQPSK	--		MAX_LEN_DQPSK(5:0)					

Name: MAX_LEN_DQPSK

Address: 0xFECA

Access: Read/write

Reset: 0xFF

- **--:** Reserved bits
- **MAX_LEN_DBPSK:** This register sets the maximum length, measured in OFDM symbols, that the system allows to receive when working with DQPSK modulation and no Viterbi encoding.

  If a message in such modulation/encoding is received and its LEN field indicates a length above the threshold defined by MAX_LEN_DQPSK value, the message will be discarded.

13.5.14 MAX_LEN_DQPSK_VTB Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
MAX_LEN_DQPSK_VTB	--		MAX_LEN_DQPSK_VTB(5:0)					

Name: MAX_LEN_DQPSK_VTB

Address: 0xFECB

Access: Read/write

Reset: 0xFF

- **--:** Reserved bits
- **MAX_LEN_DQPSK_VTB:** This register sets the maximum length, measured in OFDM symbols, that the system allows to receive when working with DQPSK modulation and Viterbi encoding.

  If a message in such modulation/encoding is received and its LEN field indicates a length above the threshold defined by MAX_LEN_DQPSK_VTB value, the message will be discarded.

13.5.15 MAX_LEN_D8PSK Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
MAX_LEN_D8PSK	--		MAX_LEN_D8PSK(5:0)					

Name: MAX_LEN_D8PSK

Address: 0xFECC

Access: Read/write

Reset: 0xFF

- **--:** Reserved bits
- **MAX_LEN_D8PSK:** This register sets the maximum length, measured in OFDM symbols, that the system allows to receive when working with D8PSK modulation and no Viterbi encoding.

  If a message in such modulation/encoding is received and its LEN field indicates a length above the threshold defined by MAX_LEN_D8PSK value, the message will be discarded.

13.5.16 MAX_LEN_D8PSK_VTB Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
MAX_LEN_D8PSK_VTB	--		MAX_LEN_D8PSK_VTB(5:0)					

Name: MAX_LEN_D8PSK_VTB

Address: 0xFECD

Access: Read/write

Reset: 0xFF

- **--:** Reserved bits
- **MAX_LEN_D8PSK_VTB:** This register sets the maximum length, measured in OFDM symbols, that the system allows to receive when working with D8PSK modulation and Viterbi encoding.

  If a message in such modulation/encoding is received and its LEN field indicates a length above the threshold defined by MAX_LEN_D8PSK_VTB value, the message will be discarded.

13.5.17 AES_PAD_LEN Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
AES_PAD_LEN	--				AES_PAD_LEN(3:0)			

Name: AES_PAD_LEN

Address: 0xFE25

Access: Read/write

Reset: 0x00

- **--:** Reserved bits
- **AES_PAD_LEN:** AES protocol works over 16-bytes-lenght blocks. When a block is not 16-bytes long, this register indicates the number of padding bytes to append.

  This register takes values between 0 and 15.

  In transmission, if encryption is being used, microcontroller must write the AES padding length in this register.

  In no-encrypted transmission and in reception, the value in this register is not used.

13.5.18 AES_DATA_IN Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
AES_DATA_IN	AES_DATA_IN(127:120)								@FFA0
	…							…	
	AES_DATA_IN(7: 0)								@FFAF

Name: AES_DATA_IN

Address: 0xFFA0 – 0xFFAF

Access: Read/write

Reset: 0x00, …, 0x00

- **AES_DATA_IN:** Input buffer for AES128 block.
  This buffer can be written to be encrypted/decrypted by the key in KEY_PERIPH (see 13.5.20) register.
  The resulting data could be read at AES_DATA_OUT (see 13.5.19) registers.

13.5.19 AES_DATA_OUT Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
AES_DATA_OUT	AES_DATA_OUT(127:120)								@FFB0
	…							…	
	AES_DATA_OUT(7: 0)								@FFBF

Name: AES_DATA_OUT

Address: 0xFFB0 – 0xFFBF

Access: Read only

Reset: 0x00, …, 0x00

- **AES_DATA_OUT:** Output buffer for AES128 block.

  This buffer stores the result of the encryption/decryption processing of data in AES_DATA_IN (see 13.5.18) register with the key in KEY_PERIPH (see 13.5.20) register.

13.5.20 KEY_PERIPH Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
KEY_PERIPH	KEY_PERIPH(127:120)								@FFC0
	…								
								…	
	KEY_PERIPH (7: 0)								@FFCF

Name: KEY_PERIPH

Address: 0xFFC0 – 0xFFCF

Access: Read/write

Reset:

KEY_PERIPH(127:120) :	0x00;	KEY_PERIPH(119:112) :	0x01;	KEY_PERIPH(111:104) :	0x02;
KEY_PERIPH(103:96) :	0x03;	KEY_PERIPH(95:88) :	0x04;	KEY_PERIPH(87:80) :	0x05;
KEY_PERIPH(79:72) :	0x06;	KEY_PERIPH(71:64) :	0x07;	KEY_PERIPH(63:56) :	0x08;
KEY_PERIPH(55:48) :	0x09;	KEY_PERIPH(47:40) :	0x0A;	KEY_PERIPH(39:32) :	0x0B;
KEY_PERIPH(31:24) :	0x0C;	KEY_PERIPH(23:16) :	0x0D;	KEY_PERIPH(15:8) :	0x0E;
KEY_PERIPH(7:0) :	0x0F;				

- **KEY_PERIPH:** Key for AES128 block when used as peripheral part.
  This key is used for encrypting/decrypting data in AES_DATA_IN registers.

13.5.21 KEY_PHY Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
KEY_PHY	KEY_PHY(127:120)								@FFD0
	…							…	
	KEY_PHY(7: 0)								@FFDF

Name: KEY_PHY

Address: 0xFFD0 – 0xFFDF

Access: Read/write

Reset:

KEY_PHY(127:120):	0x00;	KEY_PHY(119:112) :	0x01;	KEY_PHY(111:104) :	0x02;
KEY_PHY(103:96) :	0x03;	KEY_PHY(95:88) :	0x04;	KEY_PHY(87:80) :	0x05;
KEY_PHY(79:72) :	0x06;	KEY_PHY(71:64) :	0x07;	KEY_PHY(63:56) :	0x08;
KEY_PHY(55:48) :	0x09;	KEY_PHY(47:40) :	0x0A;	KEY_PHY(39:32) :	0x0B;
KEY_PHY(31:24) :	0x0C;	KEY_PHY(23:16) :	0x0D;	KEY_PHY(15:8) :	0x0E;
KEY_PHY(7:0) :	0x0F;				

- **KEY_PHY:** Key for AES128 block when used by the physical layer
  This key is used in real time encryption/decryption for Security Profile 1. When any of the DMA channels of the physical layer accesses to the memory, then this key and the input data are multiplexed to the AES128-core. Also output data is multiplexed in order to provide encrypted/decrypted data to the physical buffer.

13.5.22 AES_SFR Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
AES_SFR	--					READY	START	CIPHER

Name: AES_SFR

Address: 0xFFE0

Access: Read/write

Reset: 0x00

- --: Reserved bits.
- **READY:** Flag to indicate encryption/decryption process completion.
  When the encryption/decryption has been completed, this flag is set to '1'.
  This flag is automatically cleared when an encryption/decryption process begins.
- **START:** When this bit is set to '1', the encryption/decryption process is triggered.
  If encryption/decryption starts successfully, then this bit is automatically cleared to '0'.
- **CIPHER:** This field indicates if data must be encrypted or decrypted.
  - '0' - Decryption mode
  - '1' - Encryption mode

14. ATPL210 PRIME PHY Layer

14.1 ATPL210 PHY Layer

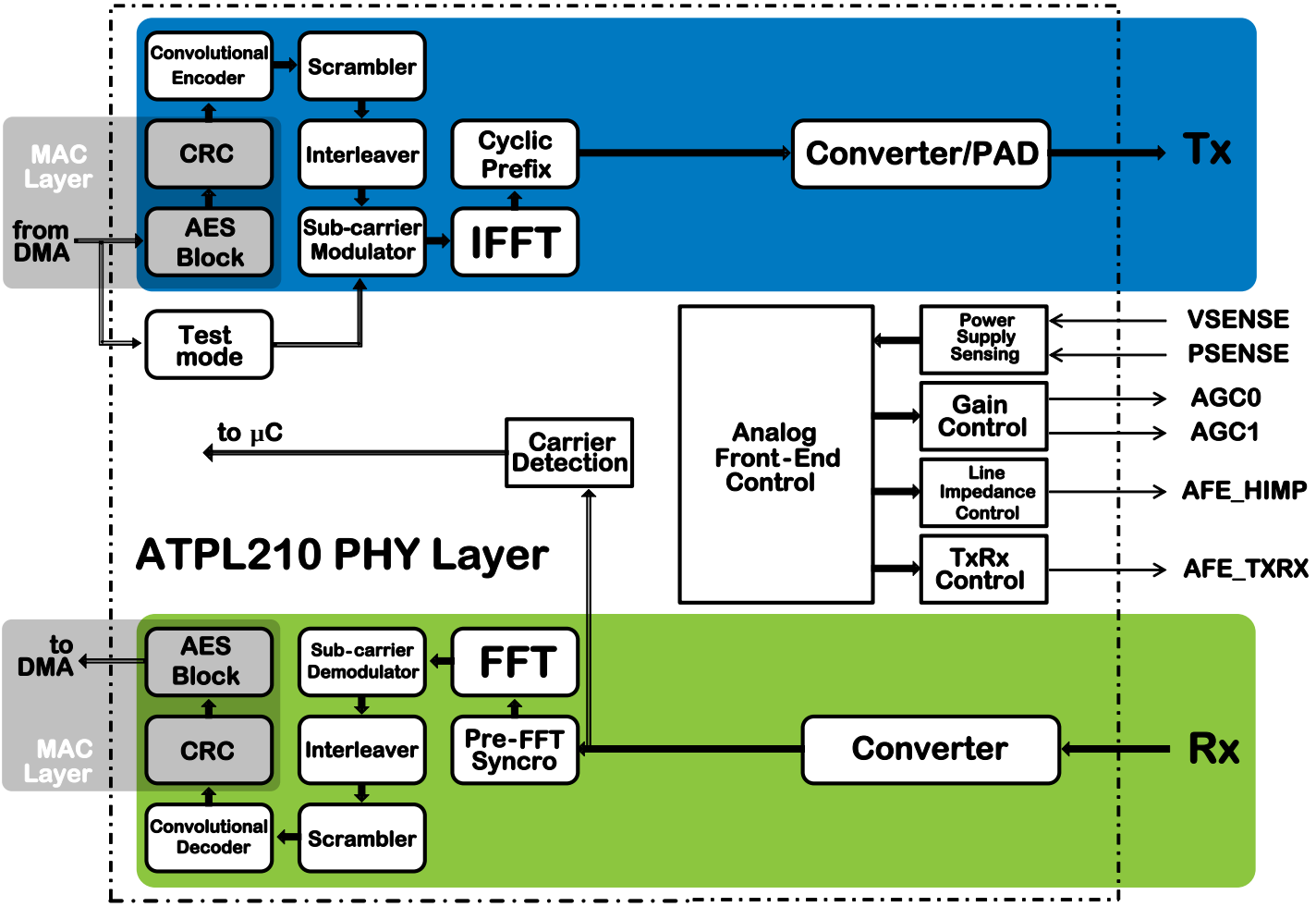
The physical layer of ATPL210 consists of a hardware implementation of the PRIME Physical Layer Entity, which is an Orthogonal Frequency Division Multiplexing (OFDM) system in the CENELEC A-band. This PHY layer transmits and receives MPDUs (MAC Protocol Data Unit) between neighbor nodes.

From the transmission point of view, the PHY layer receives its inputs from the MAC (Medium Access Control) layer, via DMA. At the end of transmission branch, data is output to the physical channel.

On the reception side, the PHY layer receives its inputs from the physical channel, and at the end of reception branch, the data flows to the MAC layer, via DMA.

A PHY layer block diagram is shown below:

Figure 14-1. ATPL210 PHY Layer Block Diagram



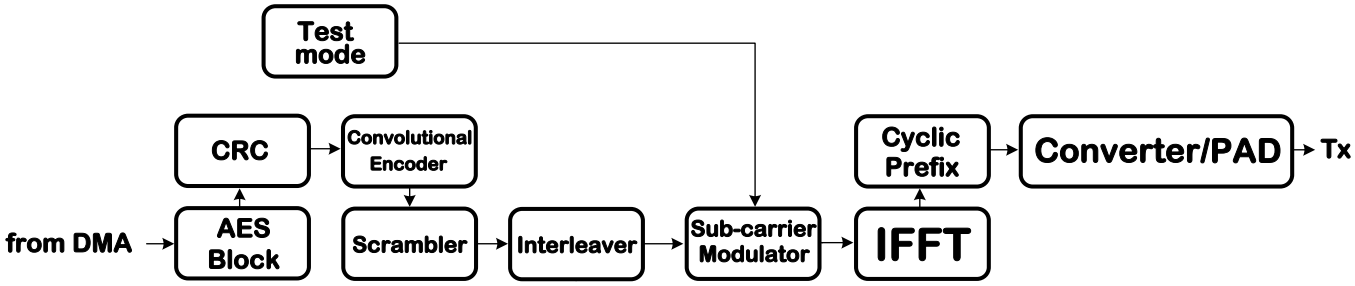
The diagram can be divided in four sub-blocks: Transmission branch, Emission branch, Analog Front End control and Carrier Detection.

14.1.2 Transmission and Reception branches

Phy layer takes data to be sent from dedicated DMA channel (PHY_TX). 128-bit AES encryption is done “on the fly”, and the Cyclic Redundancy Check (CRC) fields are hardware-generated in real time. These CRCs are properly appended to the transmission data. The rest of the chain is hardware-wired, and performs automatically all the tasks needed to send data according to PRIME specifications.

In Figure 14-2, the block diagram of the transmission branch Is shown.

Figure 14-2. Transmission branch



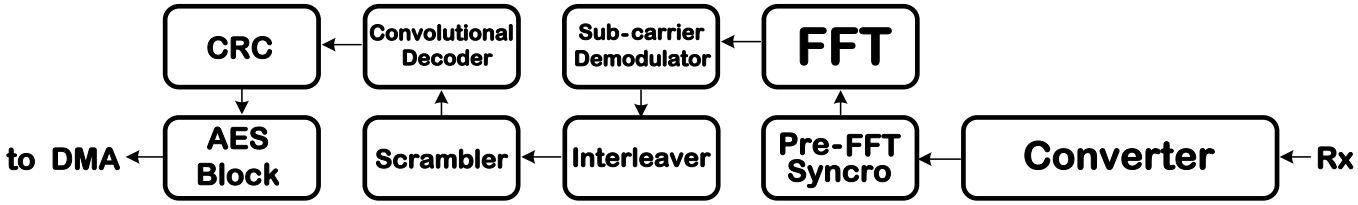
The output is differentially modulated using a BPSK/DQPSK/D8PSK scheme. After modulation, IFFT (Inverse Fourier Transform) block and cyclic prefix block allows to implement an OFDM scheme.

A Converter and a Power Amplifier Driver is the last block in the transmission branch. This block is responsible for adjusting the signal to reach the best transmission efficiency, thus reducing consumption and power dissipation.

Test mode: *When selected, test mode injects data directly to Sub-carrier modulation block. When in test mode, data can be injected continuously to the line using only a set of selected frequencies, in order to test channel behavior.*

The reception branch performs automatically all the tasks needed to process received data. Phy layer delivers data to MAC layer through the dedicated DMA channel (PHY_RX).

Figure 14-3. Reception branch



14.1.3 Carrier Detection

Looking for an easy detection of incoming messages, PRIME specification defines a chirp signal located at the beginning of the PRIME frames devised to ease synchronization in the receptor. By means of detection techniques, the receiver can know accurately when the chirp has been completely received and then the correct instant when the frame begins.

Before starting a transmission, it is also necessary to use carrier detection in order to check if another device is already emitting, thus avoiding collisions. If any device is emitting, the carrier detection triggers a microcontroller interruption and sets an internal flag, thus the transmission will be stopped.

The main drawback of this process is that chirp signal length (2.4 milliseconds) is not short enough to guarantee very low collision ratio.

To improve this drawback, the ADD1320 PLC modem implements two different algorithms to detect the carrier as soon as possible, aiming to reduce collisions and improving the medium access behavior. By these early detection techniques, the system achieves low collision ratio, and the communication throughput increases significantly.

14.1.4 Analog Front End control

The Phy layer controls the Analog Front End by means of four sub-blocks:

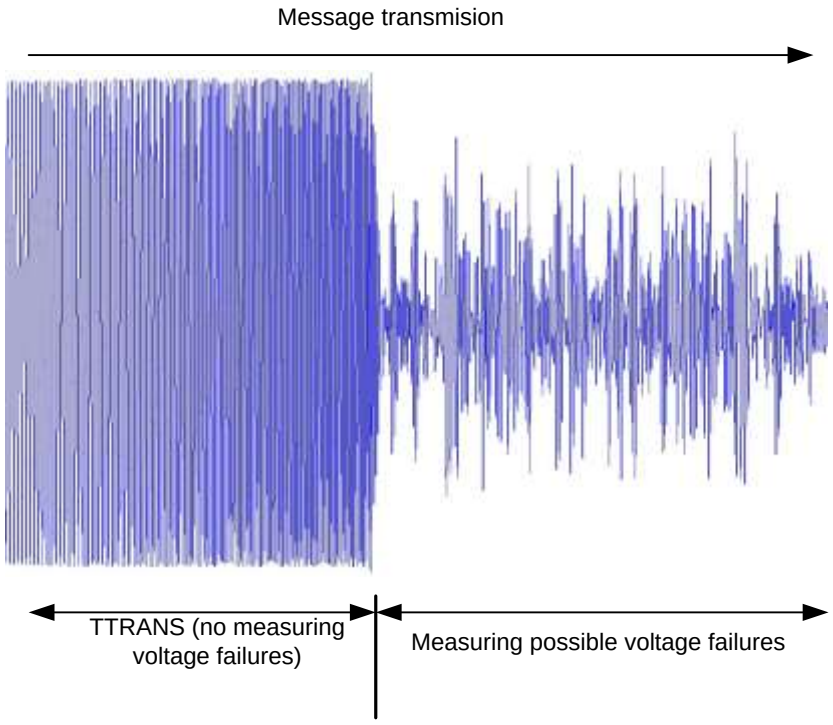
- Power Supply sensing
- Gain control
- Line Impedance control
- TxRx control

14.1.4.1 Power Supply Sensing: VSENSE and PSENSE

The power supply is continuously monitored to avoid power supply failures that could damage the supply device. This block senses the power channel using two different inputs:

- **VSENSE:** VSENSE detects whether voltage falls below 3.3v during a number of cycles while a message is being transmitted. This measurement is done after a transitory guard time (TTRANS in figure below). If a Voltage failure occurs, the transmission is shut down and sending messages again will be not possible if an internal flag (VFAILURE) is not previously cleared.

Figure 14-4. Transitory guard time in message transmission



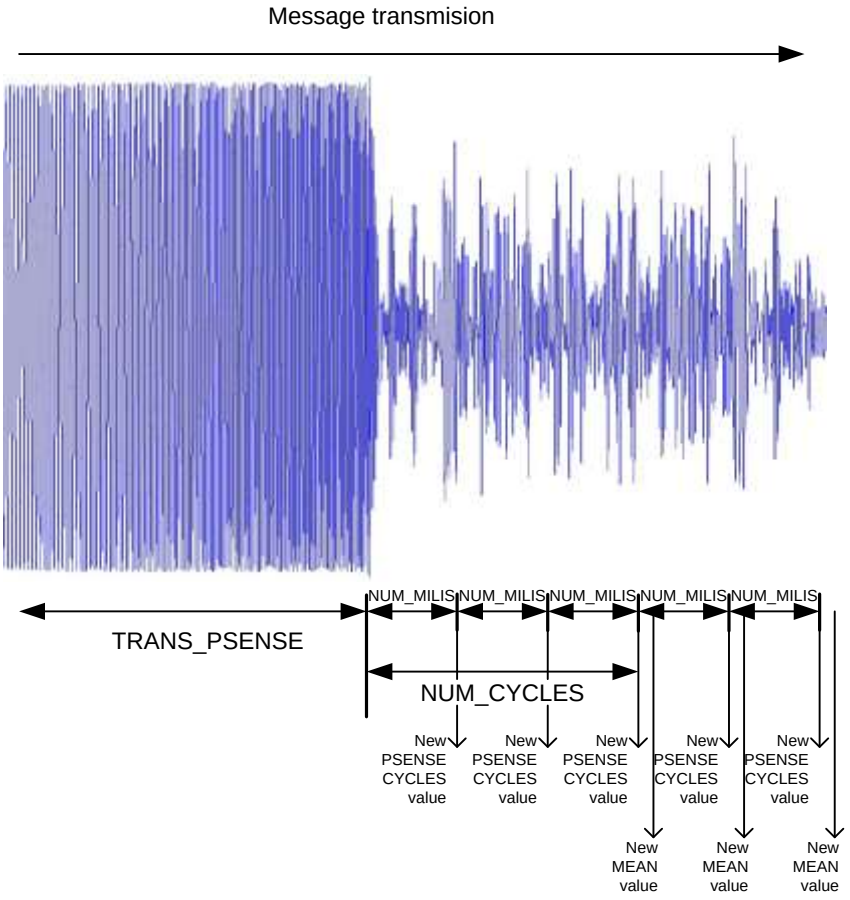
- **PSENSE:** PSENSE measures the power source current consumption, shutting down the transmission if the consumption exceeds a defined threshold (stored in MAXPOT phy layer registers, see 14.5.34).

  This measurement is done after a transitory guard time. As the current measurement varies over time, an averaging is done taking into account an average parameter (Alpha), a configurable number of cycles (NUMCYCLES, see 14.5.35) and a configurable length of each cycle (A_NUMMILIS, see 14.5.36).

If a power failure occurs, the transmission is shut down and sending messages again will be not possible if an internal flag (PFAILURE, see 14.5.22) is not previously cleared.

The system considers that a power failure has occurs when the value read from MEAN registers (see 14.5.30) is above the user-definable value stored in MAXPOT registers.

Figure 14-5. PSENSE parameters



Psense and Vsense configurations parameters are automatically set by the Phy layer.

See related peripheral registers for more information about Psense and Vsense.

14.1.4.2 Gain Control

This block implements two Automatic Gain Control outputs to adjust the received signal level to a suitable range. Both of them are set to '1' when the received signal is above two system thresholds in order to activate external attenuators placed in the external analog front end.

The value of these outputs is set during the beginning of a received message and is hold until the end of the message.

AGC0 and AGC1 follow different algorithms, thus using both of them ensures a more accurate gain control.

See AGC_CONFIG Register for information about AGC configuration.

14.1.4.3 Line Impedance Control

This block modifies the configuration of the Analog Front End by means of AFE-HIMP output. When working with a suitable external configuration, the system can change the filter conditions in order to adjust its behavior to the line impedance values. See last ATPL210 reference design for further information about Line Impedance topologies.

14.1.4.4 TxRx Control

This block modifies the configuration of the Analog Front End by means of AFE-TXRX output. Thus is possible to change filter conditions between transmission/reception.

See reference design for further information about TxRx control.

14.2 PHY parameters

As described below, the PHY layer is specified by certain main parameters, which are fixed for each specific constellation/coding combination. These parameters have to be identical in a network in order to achieve compatibility.

Table 14-1. PRIME Phy main parameters

PRIME Phy parameter	Value
Base Band Clock (Hz)	250000
Subcarrier spacing (Hz)	488,28125
Number of data subcarriers	84 (header), 96 (payload)
Number of pilot subcarriers	13 (header), 1 (payload)
FFT interval (samples)	512
FFT interval (µs)	2048
Cyclic Prefix (samples)	48
Cyclic Prefix (µs)	192
Symbol interval (samples)	560
Symbol interval (µs)	2240
Preamble period (µs)	2048

Table 14-2 shows the PHY data rate during payload transmission, and maximum MSDU length for various modulation and coding combinations

Table 14-2. Phy parameters depending on the modulation

	DBPSK		DQPSK		D8PSK	
Convolutional Code (1/2)	**On**	**Off**	**On**	**Off**	**On**	**Off**
Information bits per subcarrier	0,5	1	1	2	1,5	3
Information bits per OFDM symbol	48	96	96	192	144	288
Raw data rate (kbps approx)	21,4	42,9	42,9	85,7	64,3	128,6
MAX MSDU length with 63 symbols (bits)	3016	6048	6040	12096	9064	18144

Table 14-3 shows the modulation and coding scheme and the size of the header portion of the PHY frame

Table 14-3. Header parameters

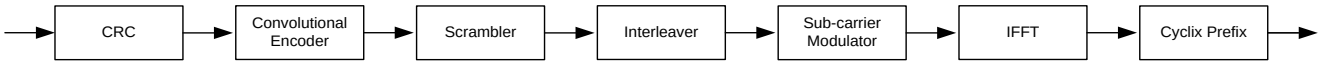
	DBPSK
Convolutional Code (1/2)	On
Information bits per subcarrier	0,5
Information bits per OFDM symbol	42

All the parameters of the physical layer such as the base band clock, subcarrier spacing, number of subcarriers...; are defined in PRIME Specification, and have to be identical in a network in order to achieve compatibility.

14.3 PHY Protocol Data Unit (PPDU) Format

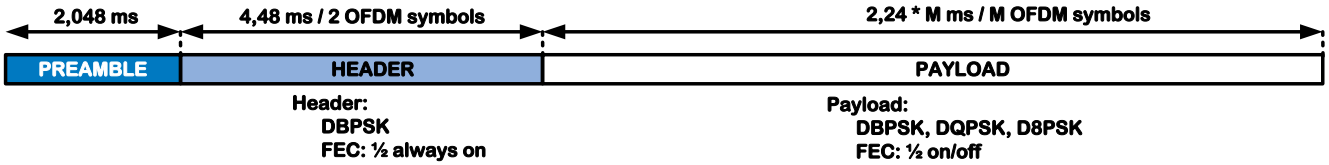
Figure 14-6 shows how OFDM symbols are transmitted in a PPDU (Physical layer Protocol Data Unit). The preamble is used at the beginning of every PPDU for synchronization purposes.

Figure 14-6. PHY layer transmitter block diagram



Phy layer adaptively modifies attenuation values applied to the whole signal. Also, additional attenuations are applied to the chirp section of the signal (preamble) and to the rest of the signal itself (header+payload), to smoothly adapt amplitude values and transitions.

Figure 14-7. PPDU OFDM symbols and duration



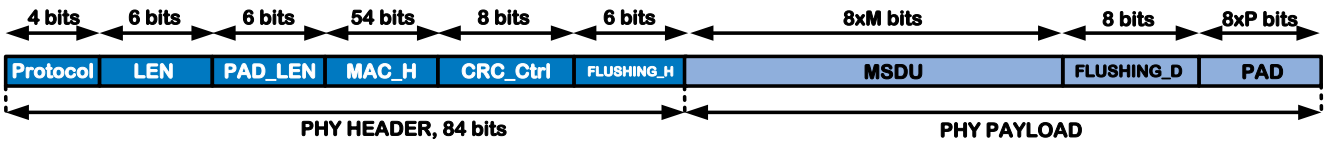
14.4 PHY Service Specification

There is an interface specified in PRIME for the PHY layer, with several primitives relative to both data and control planes.

PHY layer has a single 20-bit free-running clock measured in 10µs steps. Time measured by this clock is the one to be used in some PHY primitives to indicate a specific instant in time.

ATPL210A includes a hardware implementation of this clock, which consists of a 20-bit register. This register is read-only and it can be accessed as a 32-bit variable by the ADD8051C3A microcontroller.

Figure 14-8. Header and payload structure



Prime specifies a complete set of primitives to manage the PHY Layer, and the PHY-SAP (PHY Service Access Point) from MAC layer. Atmel PRIME stack integrates all this functions, making them transparent to the final user and simplifying the management.

14.5 PHY Layer registers

14.5.1 PHY_SFR Register

This register is described in 12.11.1, DMA Configuration Registers section.

14.5.2 SYS_CONFIG Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
SYS_CONFIG	--			CONV_PD	PHY_PD	PHY_ERR_EN	PHY_ERR	PHY_RST

Name: SYS_CONFIG

Address: 0xFE2C

Access: Read/write

Reset: 0x04

- **--:** Reserved bits
- **CONV_PD:** Converter Power Down

  Microcontroller can activate internal converter power down mode by setting this bit. When internal converter is in power down mode, the system is unable to receive.

  This bit is high-level active.
- **PHY_PD:** PHY Power Down

  This bit shuts down Physical Layer clock. When in PHY power down mode, all the system blocks involved in communication remain inactive. Thus, the system will be unable to transmit or receive. The next sequence must be respected to ensure proper power down:

  Setting PHY power down mode

  1-Set Physical Layer reset (SYS_CONFIG(0)), PHY_RST='1'

  2-Set CONV_PD and PHY_PD fields

  Exiting PHY power down mode

  1-Clear CONV_PD and PHY_PD fields

  2-Clear Physical Layer reset (SYS_CONFIG(0)), PHY_RST='0'

  This bit is high-level active.
- **PHY_ERR_EN:**Physical Layer Watchdog enable

  This bit enables or disables Physical layer watchdog. Physical layer watchdog is enabled by default.

  This bit is high-level active.
- **PHY_EN:** Physical Layer Error Flag

  This flag indicates if a Physical layer error has occurred. Physical layer watchdog has a 200milliseconds sampling period. When Physical layer detects an error, it activates the Physical layer interrupt and this flag is set.

  To restore situation, microcontroller must reset Physical layer by means of PHY_RST bit (SYS_CONFIG(0)).
- **PHY_RST:** Physical Layer Reset

  This bit resets the Physical layer. To perform a Physical layer reset cycle, microcontroller must set this bit to '1' and then must clear it to '0'.

14.5.3 PHY_CONFIG Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
PHY_CONFIG	--		**CINR_MODE**	**PAD_LEN_AC**	**AES_EN**	**CD_MOD1_EN**	**CD_MOD2_DET**	**MAC_EN**

Name: PHY_CONFIG

Address: 0xFE68

Access: Read/write

Reset: 0x1F

- **--:** Reserved bits
- **CINR_MODE:** Carrier to Interference + Noise Ratio mode

  This bit enables/disables CINR mode when set to '1'.
  - '0': CINR mode disabled.
  - '1': CINR mode enabled.
- **PAD_LEN_AC:** This field allows the system to work with two different representations of the Phy header PAD_LEN field (PAD_LEN represented before coding or PAD_LEN represented after coding).
  - '0': PAD_LEN field in PHY header is represented before coding. This is the suitable value to fulfill PRIME specification.
  - '1': PAD_LEN field in PHY header is represented after coding.
- **AES_EN:** This field enables/disables "on the fly" AES encryption and decryption by hardware.
  - '0': "on the fly" AES encryption/decryption disabled.
  - '1': "on the fly" AES encryption/decryption enabled.
- **CD_MOD1_EN:** This field enables/disables Carrier Detection mode 1.
  - '0': Carrier Detection mode 1 disabled.
  - '1': Carrier Detection mode 1 enabled.
- **CD_MOD2_DET:** This field enables/disables Carrier Detection mode 2.
  - '0': Carrier Detection mode 2 disabled.
  - '1': Carrier Detection mode 2 enabled.
- **MAC_EN**: This field enables/disables CRC processing by hardware.
  - '0': CRC processing disabled.
  - '1': CRC processing enabled.

14.5.4 ATTENUATION Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
ATTENUATION	ATTENUATION(7:0)							

Name: ATTENUATION

Address: 0xFE24

Access: Read/write

Reset: 0xFF

- **ATTENUATION:** Global attenuation for the transmitted signal (chirp+signal). The 16-bit signal level is multiplied by this 8-bit value and the result is truncated to 16 bits.
  Attenuation value = 0xFF → the transmitted signal amplitude is not attenuated.
  Attenuation value = 0x00 → the transmitted signal amplitude is nullified.

14.5.5 ATT_CHIRP Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
ATT_CHIRP	ATT_CHIRP(7:0)							

Name: ATT_CHIRP

Address: 0xFE9B

Access: Read/write

Reset: 0xFF

- **ATT_CHIRP:** This register stores the attenuation value for the chirp. The 16-bit chirp data is multiplied with this 8-bit value and the 24-bit result is truncated to 16 bits.
  Attenuation value = 0xFF → the chirp amplitude is not attenuated
  Attenuation value = 0x00 → the chirp amplitude is nullified

14.5.6 ATT_SIGNAL Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
ATT_SIGNAL	ATT_SIGNAL(7:0)							

Name: ATT_SIGNAL

Address: 0xFE9C

Access: Read/write

Reset: 0xFF

- **ATT_SIGNAL:** This register stores the attenuation value for the signal without the chirp section. The 16-bit chirp data is multiplied with this 8-bit value and the 24-bit result is truncated to 16 bits.
  Attenuation value = 0xFF → the signal amplitude is not attenuated
  Attenuation value = 0x00 → the signal amplitude is nullified

14.5.7 TX_TIME Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
TX_TIME	TX_TIME(19:12)								@0xFE26
	TX_TIME(11:4)								@0xFE27
	TX_TIME(3:0)				“0000”				@0xFE28
	“00000000”								@0xFE29

Name: TX_TIME

Address: 0xFE26 – 0xFE29

Access: Read/write

Reset: 0x00, …, 0x00;

- **TX_TIME:** This 20-bit value sets the time instant when the MPDU (MAC Protocol Data Unit) has to be transmitted. The time is expressed in 10µs steps.

  When writing a new value to TX_TIME register, a specific writing order must be taken, always from the most significant byte (TX_TIME(19:12) at address 0xFE26) to the least significant byte (TX_TIME(3:0) at address 0xFE28), and it is required to write the 3 bytes to avoid wrong time comparisons in transmission.

  The 20-bit TX_TIME value is managed by the microcontroller as a 4-byte variable. The TX_TIME value is aligned to the 20 most significant bits, being the 12 least significant bits padded with zeros.

  This register is used by the physical layer for being in accordance with PRIME specifications about transmission time (see PRIME spec.)

  Note: TXRX bit (PHY_SFR(2)) has to be cleared to ‘0’ in order to init transmission. Once this bit has been cleared, the transmission will start when TIMER_BEACON_REF value is equal to TX_TIME.

14.5.8 TIMER_FRAME Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
TIMER_FRAME	TIMER_FRAME(19:12)								@0xFE2D
	TIMER_FRAME(11:4)								@0xFE2E
	TIMER_FRAME(3:0)				“0000”				@0xFE2F
	“00000000”								@0xFE30

Name: TIMER_FRAME

Address: 0xFE2D – 0xFE30

Access: Read only

Reset: 0x00, …, 0x00;

- **TIMER_FRAME:** Time of receipt of the preamble associated with the PSDU (PHY Service Data Unit).

  It is expressed in 10µs steps and is taken from the physical layer timer TIMER_BEACON_REF.

  It is set by hardware and is a read-only register.

  This register is used by the physical layer for being in accordance with PRIME specification about reception time (see PRIME specification).

  The 20-bit TIMER_FRAME value is managed by the microcontroller as a 4-byte variable. The TIMER_FRAME value is aligned to the 20 most significant bits, being the 12 least significant bits padded with zeros. This simplifies arithmetic calculations with time values.

14.5.9 TIMER_BEACON_REF Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
TIMER_BEACON_REF	TIMER_BEACON_REF(19:12)								@0xFE47
	TIMER_BEACON_REF (11:4)								@0xFE48
	TIMER_BEACON_REF (3:0)				“0000”				@0xFE49
	“00000000”								@0xFE4A

Name: TIMER_BEACON_REF

Address: 0xFE47 – 0xFE4A

Access: Read only

Reset: 0x00, …, 0x00;

- **TIMER_BEACON_REF:** Timer for the physical layer, which consists of a single 20-bit free-running clock measured in 10µs steps.

  It indefinitely increases a unit each 10 microseconds from 0 to 1048575, overflowing back to 0.

  It is set by hardware and is a read-only register.

  This register is used by the physical layer for being in accordance with PRIME specification. It is reserved 32-bit in data memory to be able to declare as 32-bit variable. The 20-bit register MSB is aligned to the 32-bit variable MSB, in order to simplify arithmetic calculations with time values.

14.5.10 RX_LEVEL Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
RX_LEVEL	RX_LEVEL(15:8)								@0xFE31
	RX_LEVEL(7:0)								@0xFE32

Name: RX_LEVEL

Address: 0xFE31 – 0xFE32

Access: Read only

Reset: 0x00; 0x00

- **RX_LEVEL:** These registers store the autocorrelation level of the chirp signal.
  When the reception process has started, these registers are set by hardware.

14.5.11 RSSI_MIN Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
RSSI_MIN	RSSI_MIN(7:0)							

Name: RSSI_MIN

Address: 0xFE33

Access: Read only

Reset: 0xFF

- **RSSI_MIN:** Received Signal Strength Indication Min
  This register stores the minimum RSSI value measured in the last message received.
  The measurement is done at symbol level.
  The value is stored in ¼dB steps

14.5.12 RSSI_AVG Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
RSSI_AVG	RSSI_AVG(7:0)							

Name: RSSI_AVG

Address: 0xFE34

Access: Read only

Reset: 0x00

- **RSSI_AVG:** Received Signal Strength Indication Average

  This register stores the average RSSI value measured in the last message received.

  The measurement is done at symbol level.

  The value is stored in ¼dB steps

14.5.13 RSSI_MAX Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
RSSI_MAX	RSSI_MAX(7:0)								0xFE35

Name: RSSI_MAX

Address: 0xFE35

Access: Read only

Reset: 0x00

- **RSSI_MAX:** Received Signal Strength Indication Max

  This register stores the maximum RSSI value measured in the last message received.

  The measurement is done at symbol level.

  The value is stored in ¼dB steps

14.5.14 CINR_MIN Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
CINR_MIN	CINR_MIN(7:0)								0xFE38

Name: CINR_MIN

Address: 0xFE38

Access: Read only

Reset: 0xFF

- **CINR_MIN:** Carrier to Interference + Noise ratio Min

  This register stores the minimum CINR value measured in the last message received.

  In order to calculate CINR properly, the algorithm takes beacon-type messages as a reference, since this message type allows knowing its content beforehand.

  The system uses a table that must be loaded with the beacon data to be received, so CINR mode must be activated (see PHY_CONFIG register) and the same procedure used to send beacons must be followed. As CINR mode is activated, physical layer will load the message in the table instead of sending it (table load time is in the order of microseconds, and is much shorter than the one used to send the message).

  Once the table is loaded, CINR must be disabled, and next messages CINR will be calculated taken the beacon loaded in the table as reference.

  The measurement is done at symbol level.

  The value is stored in ¼dB steps.

14.5.15 CINR_AVG Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
CINR_AVG	CINR_AVG(7:0)								0xFE39

Name: CINR_AVG

Address: 0xFE39

Access: Read only

Reset: 0x00

- **CINR_AVG:** Carrier to Interference + Noise ratio Average

  This register stores the average CINR measured in the last message received.

  In order to calculate CINR properly, the algorithm takes beacon-type messages as a reference, since this message type allows knowing its content beforehand.

  The system uses a table that must be loaded with the beacon data to be received, so CINR mode must be activated (see PHY_CONFIG register) and the same procedure used to send beacons must be followed. As CINR mode is activated, physical layer will load the message in the table instead of sending it (table load time is in the order of microseconds, and is much shorter than the one used to send the message).

  Once the table is loaded, CINR must be disabled, and next messages CINR will be calculated taken the beacon loaded in the table as reference

  The measurement is done at symbol level.

  The value is stored in ¼dB steps.

14.5.16 CINR_MAX Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
CINR_MAX	CINR_MAX(7:0)								0xFE3A

Name: CINR_MAX

Address: 0xFE3A

Access: Read only

Reset: 0x00

- **CINR_MAX:** Carrier to Interference + Noise ratio Max

  This register stores the maximum CINR value measured in the last message received.

  In order to calculate CINR properly, the algorithm takes beacon-type messages as a reference, since this message type allows knowing its content beforehand.

  The system uses a table that must be loaded with the beacon data to be received, so CINR mode must be activated (see PHY_CONFIG register) and the same procedure used to send beacons must be followed. As CINR mode is activated, physical layer will load the message in the table instead of sending it (table load time is in the order of microseconds, and is much shorter than the one used to send the message).

  Once the table is loaded, CINR must be disabled, and next messages CINR will be calculated taken the beacon loaded in the table as reference

  The measurement is done at symbol level.

  The value is stored in ¼dB steps.

14.5.17 EVM_HEADER Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
EVM_HEADER	EVM_HEADER(15:8)								@0xFE3B
	EVM_HEADER (7:0)								@0xFE3C

Name: EVM_HEADER

Address: 0xFE3B – 0xFE3C

Access: Read only

Reset: 0x00; 0x00

- **EVM_HEADER:** Header Error Vector Magnitude-

  These registers store in a 16-bit value the maximum error vector magnitude measured in the reception of a message header.

  The 7 msb (EVM_HEADER(15:9)) represent the integer part in %, being the EVM_HEADER(8:0) bits the fractional part if more precision were required.

  This register is used by the physical layer for being in accordance with PRIME specification. It is reserved 32-bit in data memory to be able to declare as 32-bit variable. The 20-bit register MSB is aligned to the 32-bit variable MSB, in order to simplify arithmetic calculations with time values.

14.5.18 EVM_PAYLOAD Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
EVM_PAYLOAD	EVM_PAYLOAD(15:8)								@0xFE3D
	EVM_PAYLOAD(7:0)								@0xFE3E

Name: EVM_PAYLOAD

Address: 0xFE3D – 0xFE3E

Access: Read only

Reset: 0x00; 0x00

- **EVM_PAYLOAD:** Payload Error Vector Magnitude-

  These registers store in a 16-bit value the maximum error vector magnitude measured in the reception of a message payload.

  The 7 msb (EVM_PAYLOAD(15:9)) represent the integer part in %, being the EVM_PAYLOAD(8:0) bits the fractional part if more precision were required.

14.5.19 EVM_HEADER_ACUM Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
EVM_HEADER_ACUM	EVM_HEADER_ACUM(19:12)								@0xFE3F
	EVM_HEADER_ACUM (11:4)								@0xFE40
	EVM_HEADER_ACUM (3:0)				“0000”				@0xFE41
	“00000000”								@0xFE42

Name: EVM_HEADER_ACUM

Address: 0xFE3F – 0xFE42

Access: Read only

Reset: 0x00, …, 0x00;

- **EVM_HEADER_ACUM:** Header Total Error Vector Magnitude Accumulator

  When receiving an OFDM symbol, the sum total of all its individual carriers EVMs is calculated in order to further calculate the average EVM value. These registers store the maximum sum total between the two OFDM symbols received in a message header.

14.5.20 EVM_PAYLOAD_ACUM Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
EVM_PAYLOAD_ACUM	EVM_PAYLOAD_ACUM(19:12)								@0xFE43
	EVM_PAYLOAD_ACUM(11:4)								@0xFE44
	EVM_PAYLOAD_ACUM(3:0)				“0000”				@0xFE45
	“00000000”								@0xFE46

Name: EVM_PAYLOAD_ACUM

Address: 0xFE43 – 0xFE46

Access: Read only

Reset: 0x00, …, 0x00;

- **EVM_PAYLOAD_ACUM:** Payload Total Error Vector Magnitude Accumulator

  When receiving an OFDM symbol, the sum total of all its individual carriers EVMs is calculated in order to further calculate the average EVM value. These registers store the maximum sum total between all the OFDM symbols received in a message payload.

14.5.21 RMS_CALC Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
RMS_CALC	RMS_CALC(7:0)							

Name: RMS_CALC

Address: 0xFE58

Access: Read only

Reset: 0x00

- **RMS_CALC:** This register stores an 8-bit value which magnitude is proportional to the emitted signal amplitude.

  By measuring the amplitude of the emitted signal, the hardware can estimate the power line input impedance. Thus hardware can adjust emission configuration appropriately.

14.5.22 VSENSE_CONFIG Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
VSENSE_CONFIG	--				PFAILURE	PSENSE_SOFT	VFAILURE	VSENSE_EN

Name: VSENSE_CONFIG

Address: 0xFE59

Access: Read only

Reset: 0x00

- **PFAILURE:** Power Failure Flag

  This flag is set when a power failure occurs. The transmission is stopped and a new transmission in not possible if this flag is not cleared previously.

  When a power failure occurs, a consideration about decreasing voltage amplitude in the source should be taken.

  This flag must be cleared by software.

- **PSENSE_SOFT:** Current measurement is done every time a transmission takes place. With PSENSE_SOFT the system can force a continuous current measurement, including both idle and transmission states.
  - '0': Current consumption is measured every time a transmission begins (after a guard time defined by TRANS_PSENSE). NUMMILIS, NUMCYCLES and TRANS_PSENSE values must be taken into account to accurate PSENSE measurements. This is the default mode and it is the expected one when ATPL210 is working.
  - '1': Current consumption is measured both in idle and transmission states. This mode is useful for design purposes, in order to find suitable values for the current threshold (MAXPOT registers) depending on the external net requirements.
- **VFAILURE:** Voltage Failure Flag

  This flag is set to '1' when a voltage failure occurs. The transmission is stopped and a new transmission is not possible if this flag is not cleared previously.

  When a voltage failure occurs, a consideration about decreasing voltage amplitude in the source should be taken.

  This flag must be cleared by software.

- **VSENSE_EN:**VSENSE enable

  This bit enables VSENSE.
  - '0': VSENSE disabled (default).
  - '1': VSENSE enabled.

14.5.23 NUM_FAILS Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
NUM_FAILS	NUM_FAILS(7:0)							

Name: NUM_FAILS

Address: 0xFE5A

Access: Read/write

Reset: 0x02

- **NUM_FAILS:** This register stores the number of 50 ns cycles (clk=20MHz) during which a voltage failure must be detected before shutting off the transmission and setting VFAILURE flag.

  This detection shall be done after a guard period set by TTRANS from the beginning of the transmission.

  Default value: 0x02 → 2 * 50 = 100ns

14.5.24 TTRANS Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
TTRANS	TTRANS(7:0)							

Name: TTRANS

Address: 0xFE5B

Access: Read/write

Reset: 0x2D

- **TTRANS:** This register stores the number of 50 µs cycles (clk=20MHz) to wait from the beginning of the transmission before looking for a possible voltage failure.

  Default value: 0x2D → 45 * 50 = 2.25ms (Thus, voltage failures are not expected until the end of chirp signal period)

14.5.25 AGC0_KRSSI Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
AGC0_KRSSI	AGC0_KRSSI(7:0)							

Name: AGC0_KRSSI

Address: 0xFE5C

Access: Read/write

Reset: 0x00

- **AGC0_KRSSI:** This register is used to correct RSSI (Received Signal Strength Indication) computation when Automatic Gain Control 0 (AGC0) is active.

14.5.26 AGC1 KRSSI Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
AGC1_KRSSI	AGC1_KRSSI(7:0)							

Name: AGC1_KRSSI

Address: 0xFE5D

Access: Read/write

Reset: 0x00

- **AGC1_KRSSI:** This register is used to correct RSSI (Received Signal Strength Indication) computation when Automatic Gain Control 1 (AGC1) is active.

14.5.27 ZERO_CROSS_TIME Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
ZERO_CROSS_TIME	ZERO_CROSS_TIME(19:12)								@0xFE69
	ZERO_CROSS_TIME (11:4)								@0xFE6A
	ZERO_CROSS_TIME (3:0)				“0000”				@0xFE6B
	“00000000”								@0xFE6C

Name: ZERO_CROSS_TIME

Address: 0xFE69 – 0xFE6C

Access: Read only

Reset: 0x00, …, 0x00;

- **ZERO_CROSS_TIME:** Instant in time at which the last zero-cross event took place. It is expressed in 10µs steps and may take values from 0 to 1e6 (20-bit effective).

  It is set by hardware and is a read-only register.

  This register is used by the physical layer for being in accordance with PRIME specification. It is reserved 32-bit in data memory to be able to declare as 32-bit variable. The 20-bit register MSB is aligned to the 32-bit variable MSB, in order to simplify arithmetic calculations with time values.

14.5.28 ZERO_CROSS_CONFIG Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
ZERO_CROSS_CONFIG	--					VEZC	REZC	FEZC

Name: ZERO_CROSS_CONFIG

Address: 0xFE6D

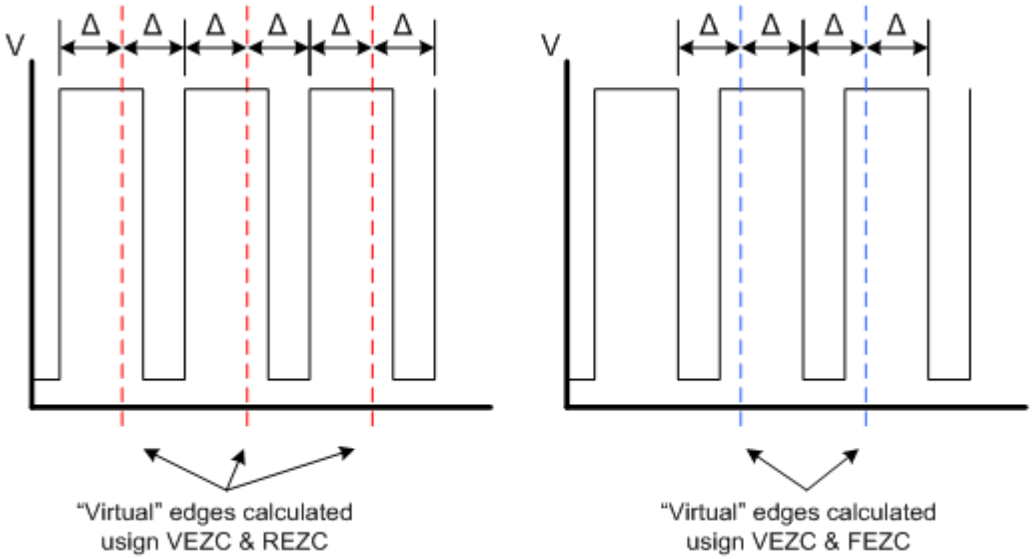
Access: Read/write

Reset: 0x06

- **--:** Reserved bits
- **VEZC:** Virtual Edge for Zero Crossing

  In this bit is equal to one, the hardware calculates the middle point between two VNR edges to calculate de zero crossing.

  This mode is used when the VNR signal duty cycle is different from 50%:



  VECZ can be used simultaneously with RECZ or FECZ.

  Using the three of them at a time is not recommended.

- **REZC**: Rising Edge for Zero Crossing

  If this bit is set to '1', the hardware uses the VNR rising edges to calculate zero-crossing.

  FEZC and REZC can be used simultaneously.

- **FEZC:** Falling Edge for Zero Crossing

  If this bit is set to '1', the hardware uses the VNR falling edges to calculate zero-crossing.

  FEZC and REZC can be used simultaneously.

14.5.29 PSENSECYCLES Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
PSENSECYCLES	--				FLAG_PSENSE	D(18:16)			@0xFE7D
	D(15:8)								@0xFE7E
	D(7:0)								@0xFE7F

Name: PSENSECYCLES

Address: 0xFE7D – 0xFE7F

Access: Read/write

Reset: 0x00, …, 0x00;

- **--:** Reserved bits
- **FLAG_SENSE:** Whenever a new power value is written in PSENSECYCLES, FLAG_PSENSE is set to'1'.

  This flag must be cleared by software
- **D(17:0):** Power supply consumption measurement

  The power supply line is sampled (=20MHz), and the number of logic '1' detected during NUMMILIS milliseconds is stored in this field in order to calculate power consumption.

  Note: The first valid value is written after NUMMILIS, and then a new valid value is written every NUMMILIS milliseconds.

  Note: Measurement is only active when a message transmission begins or PSENSE_SOFT bit is active (see Name:14.5.22)

14.5.30 MEAN Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
MEAN	--				FLAG_MEAN	D(18:16)			@0xFE80
	D(15:8)								@0xFE81
	D(7:0)								@0xFE82

Name: MEAN

Address: 0xFE80 – 0xFE82

Access: Read/write

Reset: 0x00, …, 0x00;

- **--:** Reserved bits
- **FLAG_MEAN:** Whenever a new value is written in MEAN, FLAG_MEAN is set to '1'

  This flag must be cleared by software
- **D(17:0):** This value stores the average power consumption calculated from the value in PSENSECYCLES and having into account the convergence factor "A" (see A_NUMMILIS register in 14.5.36).

  Note: The first valid value is written after NUMCYCLES*NUMMILIS, and then a new valid value is written every NUMMILIS milliseconds

14.5.31 PMAX Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
PMAX	--				FLAG_PMAX	D(18:16)			@0xFE83
	D(15:8)								@0xFE84
	D(7:0)								@0xFE85

Name: PMAX

Address: 0xFE83 – 0xFE85

Access: Read/write

Reset: 0x00, …, 0x00;

- **--:** Reserved bits
- **FLAG_PMAX:** Whenever a new value is written in PMAX, FLAG_PMAX is set to '1'.
  This flag must be cleared by software
- **D(17:0):** As described in MAXPOT register (see 14.5.34), every time the average power consumption exceeds a user defined threshold value, the current transmission is cancelled.
  PMAX register stores the average power consumption value that has risen above MAXPOT threshold.

14.5.32 TRANS_PSENSE Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
TRANS_PSENSE	TRANS_PSENSE(7:0)							

Name: TRANS_PSENSE

Address: 0xFE86

Access: Read/write

Reset: 0x2B

- **TRANS_PSENSE:** This register stores the number of 50 µs cycles to wait from the beginning of a transmission before looking for a possible power failure. This guard time is taken to avoid transient period where the measurement would be inaccurate
  Default value: 0x2B → 43 * 50 = 2.15ms

14.5.33 P_TH Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
P_TH	--					P_TH(18:16)			@0xFE87
	P_TH(15:8)								@0xFE88
	P_TH(7:0)								@0xFE89

Name: P_TH

Address: 0xFE87 – 0xFE89

Access: Read/write

Reset: 0x07, 0xFF, 0xFF.

- **--:** Reserved bits
- **P_TH:** These registers contain a user defined power threshold. When the threshold value is exceeded, a low power consumption mode is automatically activated. In this low power consumption mode, the power dissipated in the transistors decreases at the expense of distortion increasing.

14.5.34 MAXPOT Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
MAXPOT	--					MAXPOT(18:16)			@0xFE8A
	MAXPOT (15:8)								@0xFE8B
	MAXPOT (7:0)								@0xFE8C

Name: MAXPOT

Address: 0xFE8A – 0xFE8C

Access: Read/write

Reset: 0x07, 0xFF, 0xFF.

- **--:** Reserved bits
- **MAXPOT:** These registers contain a user defined power consumption threshold. When this threshold is exceeded, current transmission is cancelled.

  When the threshold is exceeded, two flags are activated:

  - POTFAILURE flag (see VSENSE_CONFIG in 14.5.22). This flag indicates that a power failure has occurred.
  - FLAG_PMAX flag (see PMAX in 14.5.31). This flag indicates that, after a power failure, the last mean power value measured has been stored in PMAX register.

  To reset both flags is enough to reset either of them, the other will be automatically reset. This will enable to start new transmissions.

14.5.35 NUMCYCLES Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
NUMCYCLES	NUMCYCLES(7:0)							

Name: NUMCYCLES

Address: 0xFE8D

Access: Read/write

Reset: 0x05

- **NUMCYCLES:** Number of cycles of measuring power before obtaining a mean value that can be taken as valid.

  Example1: If NUMCYCLES=5(cycles) and NUMMILIS=1(milliseconds), 5 power measurements will be taken during 1 millisecond each one .The first valid power measurement value will be output in the fifth millisecond.

  Example2: If NUMCYCLES=3(cycles) and NUMMILIS=20(milliseconds), 3 power measurements will be taken during 20 milliseconds each one. The first valid power measurement value will be output after 60 milliseconds.

14.5.36 A_NUMMILIS Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
NUMMILIS	--	A(1:0)		NUMMILIS(4:0)				

Name: A_NUMMILIS

Address: 0xFE8E

Access: Read/write

Reset: 0x21

- **--:** Reserved bits
- **A(1:0):** Convergence Factor

  Averaging factor that sets the convergence speed of the mean calculation algorithm.

  A=00 sets quicker convergence, while A=11 sets the slowest one. A=01,10 are intermediate values.

  Note: Power supply presents high dispersion values, so NUMMILIS value must be take into account in order to select a suitable value for A. If NUMMILIS is high, the mean value can be calculated slowly, because the averaging in being calculated over a long period of time. When NUMMILIS is low, the mean value must be calculated quickly in order to obtain more accurate values.
- **NUMMILIS(4:0):** Measurement acquisition time in milliseconds

  Stores the measurement acquisition time in milliseconds.

  Example1: If NUMCYCLES=5(cycles) and NUMMILIS=1(milliseconds), 5 power measurements will be taken during 1 millisecond each one .The first valid power measurement value will be output in the fifth millisecond.

  Example2: If NUMCYCLES=3(cycles) and NUMMILIS=20(milliseconds), 3 power measurements will be taken during 20 milliseconds each one. The first valid power measurement value will be output after 60 milliseconds.

14.5.37 EMIT_CONFIG Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
EMIT_CONFIG	--						TR_EMIT	TWO_H_BRIDGES

Name: EMIT_CONFIG

Address: 0xFE8F

Access: Read/write

Reset: 0x03

- **--:** Reserved bits
- **TR_EMIT:** Emission mode

  This bit selects the emission mode (Internal Drive or External transistors bridge).
  - '0': Emission is done by means of internal ATPL210 driver.
  - '1': Emission is done by means of external transistors (Default).
- **TWO_H_BRIDGES:** This bit selects the number of semi-H-bridges in the external interface.
  - '0': There is only one semi-H-bridge in the external interface.
  - '1': There are two semi-H-bridges in the external interface and the field HIMP (AFE_CTL register) determines which one is active (Default).

  Semi-H-Bridges must be connected following the table below

	TWO_H_BRIDGES='0'	TWO_H_BRIDGES='1'
EMIT1	P	P1
EMIT2	P	P1
EMIT3	P	P1
EMIT4	P	N1
EMIT5	P	N1
EMIT6	P	N1
EMIT7	N	P2
EMIT8	N	P2
EMIT9	N	P2
EMIT10	N	N2
EMIT11	N	N2
EMIT12	N	N2

14.5.38 AFE_CTL Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
AFE_CTL	--			HIMP	HIMP_INV	TXRX	TXRX_HARD	TXRX_INV

Name: AFE_CTL

Address: 0xFE90

Access: Read/write

Reset: 0x10

- **--:** Reserved bits
- **HIMP:** Analog Front End Impedance control bit-

  This bit selects which branch is active when working with a two half-H-bridge branches analog front end.

  - '0': "Low impedance" half-H-bridge is active (P2-N2).
  - '1': "High impedance" half-H-bridge is active (P1-N1).
- **HIMP_INV:** HIMP pin polarity control

  This field inverts the polarity of the HIMP pin output.

  Note: This field only affect to the polarity of the external pin HIMP output, the value taken from HIMP bit (AFE_CTL(4)) remains unchanged
- **TXRX:** The value stored in this bit is taken by the microcontroller in order to set the TXRX pin level.
  - '0': TXRX pin output = '0'.
  - '1': TXRX pin output = '1'.
- **TXRX_HARD:** TXRX pin control

  This field selects if the TXRX pin is software/hardware controlled.
  - '0': TXRX pin is software controlled. TXRX value is set by TXRX bit field (AFE_CTL(2)).
  - '1': TXRX pin is hardware controlled.
- **TXRX_INV:** TXRX pin polarity control

  This field inverts the polarity of the TXRX pin output

14.5.39 R Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
R1	R1(7:0)								0xFE9F
R2	R2(7:0)								0xFEA0
R3	R3(7:0)								0xFEA1
R4	R4(7:0)								0xFEA2
R5	R5(7:0)								0xFEA3
R6	R6(7:0)								0xFEA4
R7	R7(7:0)								0xFEA5
R8	R8(7:0)								0xFEA6

Name: R1 – R8

Address: 0xFE9F – 0xFEA6

Access: Read/write

Reset: 0x60; 0x60; 0x60; 0x60; 0xFF; 0xFF; 0xFF; 0xFF.

- **R:** The value in these registers strongly depends on the external circuit configuration.
  Atmel provides values to be used according with the design recommended in ATPL210 kits
  Please contact Atmel Power Line if different external configurations are going to be used

  Recommended values (according to the configuration recommended in ATPL210 kits)
  R1(7:0): 0x21
  R2(7:0): 0x20
  R3(7:0): 0x12
  R4(7:0): 0x02
  R5(7:0): 0x37
  R6(7:0): 0x77
  R7(7:0): 0x37
  R8(7:0): 0x77

  Order of precedence: In the event of a conflict between the Ri(7:0) values above and Ri(7:0) values specified in the latest documentation in an ATPL210 kit, the values in the kit documentation shall take precedence.

14.5.40 PHY_ERRORS Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
PHY_ERRORS	--			PHY_ERRORS(4:0)				

Name: PHY_ERRORS

Address: 0xFE94

Access: Read/write

Reset: 0x00

- **--:** Reserved bits
- **PHY_ERRORS:** Physical Layer Error Counter

  The system stores in these bits the number of times that a Physical layer error has occurred. Microcontroller can clear this counter to zero.

  The value stored in this register is cleared every time the register is read.

14.5.41 FFT_MODE Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
FFT_MODE	NSYM(5:0)						CONTINUOUS	TEST_MODE_EN

Name: FFT_MODE

Address: 0xFEB0

Access: Read/write

Reset: 0x00

- **NSYM:** Number of symbols to transmit

  When in continuous transmission mode, symbol data acts as a free-running buffer, increasing from 0 to NSYM-1 and overflowing back to symbol 0.
- **CONTINUOUS:** This field enables/disables continuous transmission mode.
  - '0': Continuous transmission mode disabled.
  - '1': Continuous transmission mode enabled.
- **TEST_MODE_EN:** This field enables/disables test mode
  - '0': Test mode disabled.
  - '1': Test mode enabled.

Configuration for test mode. This register is used by the physical layer to fulfill with PRIME specification (PLME_TESTMODE.request primitive and PLME_TESTMODE.confirm primitive, see PRIME specification). In this mode data provided to FFT is written in data memory at ADDR_PHY_INI_TX, codifying each value with 4 bits according to DPSK modulation mapping. The msb of the value is to indicate an input of zero when set to '1'. Each byte in data memory contains 2 input values for FFT, with the first value located at high bits. There are 97 input values for FFT, so many as the number of subcarriers, so there are 48 bytes and a half of the next byte used for codifying them. The other half of this byte (low bits) will be used for the next symbol data.

14.5.42 AGC_CONFIG Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
AGC_CONFIG	--		AGC0_POL	AGC0_VALUE	AGC0_MODE	AGC1_POL	AGC1_VALUE	AGC1_MODE

Name: AGC_CONFIG

Address: 0xFEB1

Access: Read/write

Reset: 0x24

ATPL210 has implemented two Automatic Gain Control outputs in order to adjust the received signal level to a suitable range. When in “automatic” mode, both of them are set to ‘1’ when the received signal is above a 16-bit-user-definable thresholds (AGC1_TH and AGC0_TH) in order to activate external attenuators placed in the external analog front end.

The value of these outputs is set during the beginning of a received message and is hold until the end of the message.

AGC0 and AGC1 follow different algorithms, thus using both of them ensures more accurate gain control

- --: Reserved bits
- **AGC0_POL:** AGC0 polarity

  This bit sets the polarity of the AGC0 output.

  - ‘0’: Polarity is inverted.
  - ‘1’: Polarity is not inverted (default).
- **AGC0_VALUE:** AGC0 output value-

  This bit stores the value wrote by the user to be the AGC0 output.

  This bit is only taken into account when AGC0 “forced” mode is active (AGC0_MODE=’1’).

  AGC0_POL field can invert this value.
- **AGC0_MODE:** AGC0 mode

  This bit selects which AGC0 mode is being used

  - ‘0’: “Automatic” Mode. AGC0 output will be managed by the MAC, depending on saturation detected in received signal. If saturation is detected, AGC0 output will be ‘1’. Else, AGC0 output will be ‘0’. AGC0_POL field can invert this value.

    (See SAT_TH registers in 14.5.43)
  - ‘1’: “Forced” Mode. AGC0 output will be managed by the user, according to the value wrote in AGC0_VALUE field (AGC_CONFIG(4)).
- **AGC1_POL:** AGC1 polarity

  This bit sets the polarity of the AGC1 output.

  - ‘0’: Polarity is inverted.
  - ‘1’: Polarity is not inverted (default).
- **AGC1_VALUE:** AGC1 output value-

  This bit stores the value wrote by the user to be the AGC1 output.

  This bit is only taken into account when AGC1 “forced” mode is active (AGC1_MODE=’1’).

  AGC1_POL field can invert this value.

- **AGC1_MODE:** AGC1 mode

  This bit selects which AGC1 mode is being used

  - ‘0’: “Automatic” Mode. AGC1 output will be managed by the MAC, depending on saturation detected in received signal. If saturation is detected, AGC1 output will be ‘1’. Else, AGC1 output will be ‘0’. AGC1_POL field can invert this value.

    (See SAT_TH registers in 14.5.43)
  - ‘1’: “Forced” Mode. AGC1 output will be managed by the user, according to the value wrote in AGC1_VALUE field (AGC_CONFIG(4)).

14.5.43 SAT_TH Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
SAT_TH	SAT_TH(15:8)								@0xFEB7
	SAT_TH(7:0)								@0xFEB8

Name: SAT_TH

Address: 0xFEB7 – 0xFEB8

Access: Read/write

Reset: 0x40; 0x00

- **SAT_TH:** These registers store a threshold for the PLC input-signal amplitude.

  If this threshold is exceeded, AGC thresholds (AGC0_TH and AGC1_TH) will be taken into account.

  If this threshold is not exceeded, AGC0_TH and AGC1_TH thresholds will be ignored, thus the AGC algorithm will be never triggered.

  Recommended value for Atmel reference design = 0x37AA.

14.5.44 AGC1_TH Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
AGC1_TH	AGC1_TH(15:8)								@0xFE5F
	AGC1_TH(7:0)								@0xFE60

Name: AGC1_TH

Address: 0xFE5F – 0xFE60

Access: Read/write

Reset: 0x40; 0x00

- **AGC1_TH:** AGC1 Threshold

  These registers store the 16-bit upper threshold used by the AGC1 algorithm to determine that the input signal must be attenuated.

  This threshold is only taken into account in AGC1 “automatic” mode (AGC_CONFIG.AGC1_MODE=’0’).

  This threshold is only taken into account if SAT_TH value is exceeded.

  Recommended value for Atmel reference design = 0x4A00.

14.5.45 AGC0_TH Registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0	
AGC0_TH	AGC0_TH(15:8)								@0xFEB2
	AGC0_TH(7:0)								@0xFEB3

Name: AGC0_TH

Address: 0xFEB2 – 0xFEB3

Access: Read/write

Reset: 0x10; 0x00

- **AGC0_TH:** AGC0 Threshold

  These registers store the 16-bit upper threshold used by the AGC0 algorithm to determine that the input signal must be attenuated.

  This threshold is only taken into account in AGC0 “automatic” mode (AGC_CONFIG.AGC0_MODE=’0’).

  This threshold is only taken into account if SAT_TH value is exceeded.

  Recommended value for Atmel reference design = 0x1000.

14.5.46 AGC_PADS Register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
AGC_PADS	--						P46_MODE	SWITCH_AGC

Name: AGC_PADS

Address: 0xFE61

Access: Read/write

Reset: 0x00

- --: Reserved bits
- **P46_MODE:** This field controls the P4.6/T2/AGC1 output pin (pin no.94).
  - '0': Pin no.94 works as P4.6/T2 output pin.
  - '1': Pin no.94 works as AGC1 output pin.
- **SWITCH_AGC:** This bit switches the AGC0 and AGC1 outputs.
  - '0': Not switched AGC outputs.
  - '1': Switched AGC outputs.

15. Electrical Characteristics

15.1 Absolute Maximum Ratings

Permanent device damage may occur if Absolute Maximum Ratings are exceeded. Functional operation should be restricted to the conditions given in the Recommended Operating Conditions section. Exposure to the Absolute Maximum Conditions for extended periods may affect device reliability.

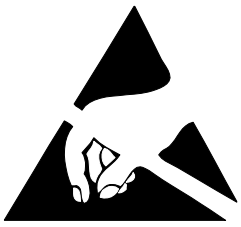
Table 15-1. ATPL210A Absolute Maximum Ratings

Parameter	Symbol	Rating	Unit
Supply Voltage	VCC	-0.5 to 4.0	V
Input Voltage	VI	-0.5 to VCC+0.5(≤4.0V)	V
Output Voltage	VO	-0.5 to VCC+0.5(<4.0V)	V
Storage Temperature	TST	-55 to 125	ºC
Junction Temperature	TJ	-40 to 125	ºC
Output Current(1)	IO	±10(2)	mA

Notes: 1. DC current that continuously flows for 10ms or more, or average DC current.

2. Applies to all the pins except EMIT pins. EMIT pins should be only used according to circuit configurations recommended by Atmel Power Line.

ATTENTION observe EDS precautions



Precautions for handling electrostatic sensitive devices should be taken into account to avoid malfunction. Charged devices and circuit boards can discharge without detection

15.2 Recommended Operating Conditions

Table 15-2. ATPL210A Recommended Operating Conditions

Parameter	Symbol	Rating			Unit
		Min	Typ	Max	
Supply Voltage	VCC	3.00	3.30	3.60	V
	VDE0	3.00	3.30	3.60	
	AVD	3.00	3.30	3.60	
Junction Temperature	TJ	-40	25	125	ºC
Ambient Temperature	TA	-40	--	85	

15.3 DC Characteristics

Table 15-3. ATPL210A DC Characteristics

Parameter	Condition	Symbol	Rating			Unit
			Min.	Typ.	Max.	
Supply Voltage		VCC	3.00	3.30	3.60	V
H-level Input Voltage (3.3v CMOS)		VIH	2.0	-	VCC+0.3	
L-level Input Voltage (3.3v CMOS)		VIL	-0.3	-	0.8	
H-level Output Voltage	3.3v I/O IOH=-100µA	VOH	VCC-0.2	-	VCC	
L-level Output Voltage	3.3v I/O IOL=100µA	VOL	0	-	0.2	
H-level Output V-I Characteristics	3.3v I/O VCC=3.3±0.3	IOH	See 15.3.2			mA
L-level Output V-I Characteristics	3.3v I/O VCC=3.3±0.3	IOL	See 15.3.2			
Internal Pull-up Resistor(1)	3.3v I/O	Rpu	10	33	80	kΩ
Internal Pull-down Resistor(1)	3.3v I/O	Rpd	10	33	80	kΩ
Junction Temperature		TJ	-40	-	125	ºC

Notes: 1. Only applicable to pins with internal pulling. See Table 2-1.

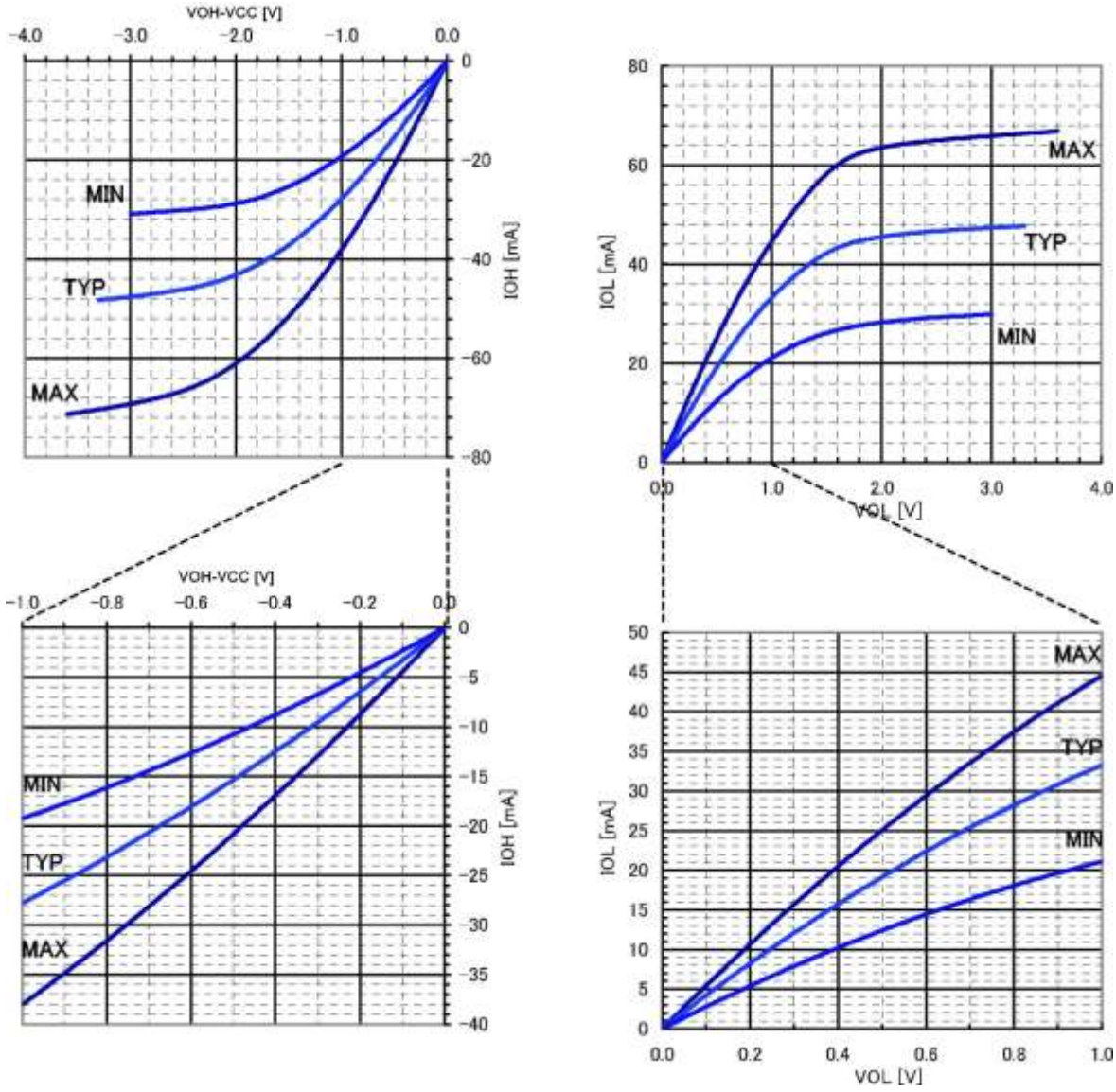
15.3.2 V-I curves

V-I Characteristics 3.3 Vstandard CMOS IO L, M type

Pins marked in Table 2-1 with Nominal Current I(mA)=±5

Condition:	MIN	Process=	Slow	Tj=	125°C	VCC=	3.0 V
	TYP	Process=Typical		Tj=	25°C	VCC=	3.3 V
	MAX	Process=Fast		Tj=	-40°C	VCC=	3.6 V

Figure 15-1. CMOS IO L and M type,V-I curves

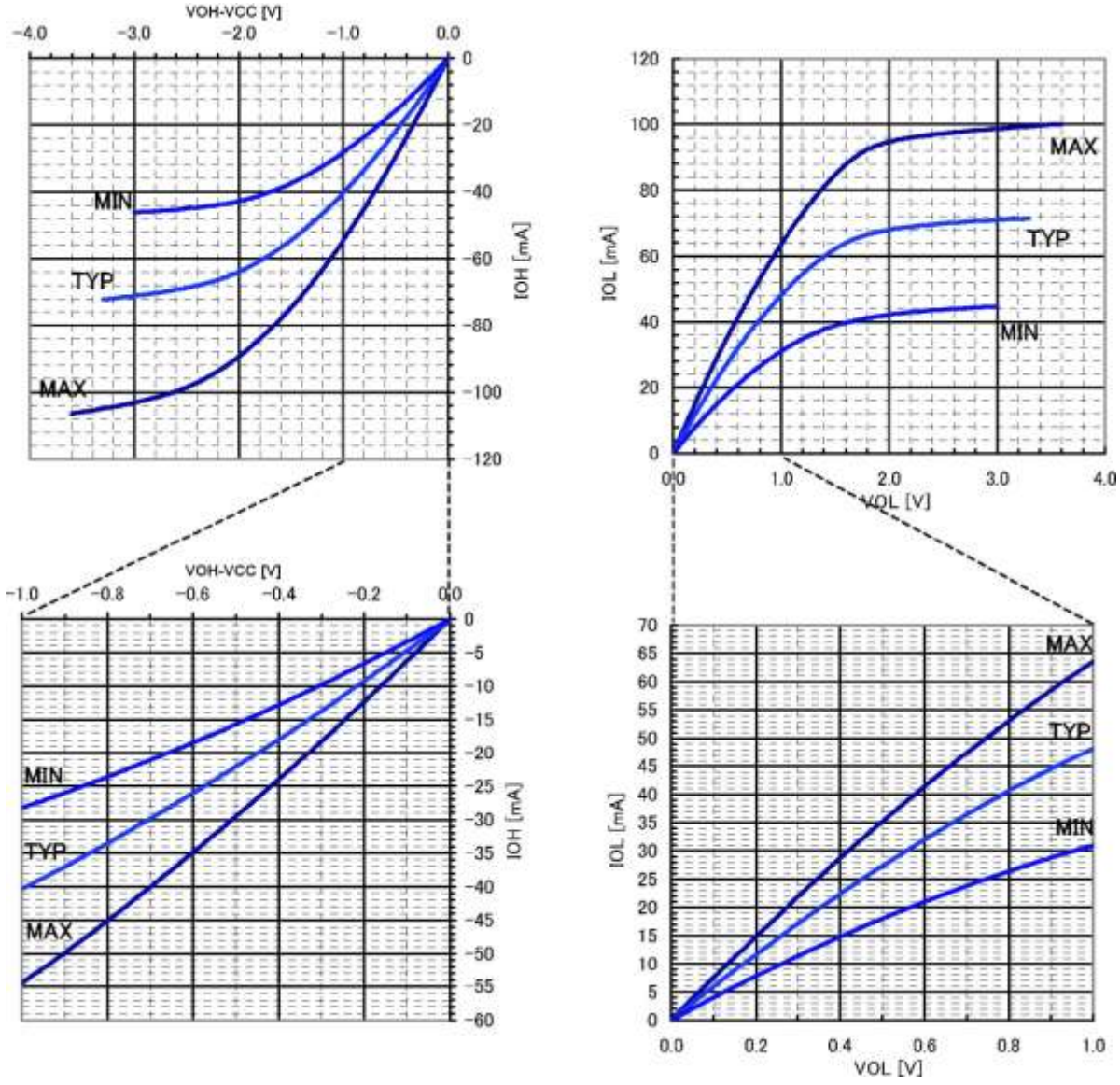


V-I Characteristics 3.3 Vstandard CMOS IO H, V type

Pins marked in Table 2-1 with Nominal Current I(mA)=±10

Condition:	MIN	Process=	Slow	Tj=	125°C	VCC=	3.0 V
	TYP	Process=Typical		Tj=	25°C	VCC=	3.3 V
	MAX	Process=Fast		Tj=	-40°C	VCC=	3.6 V

Figure 15-2. CMOS IO H and V type, V-I curves

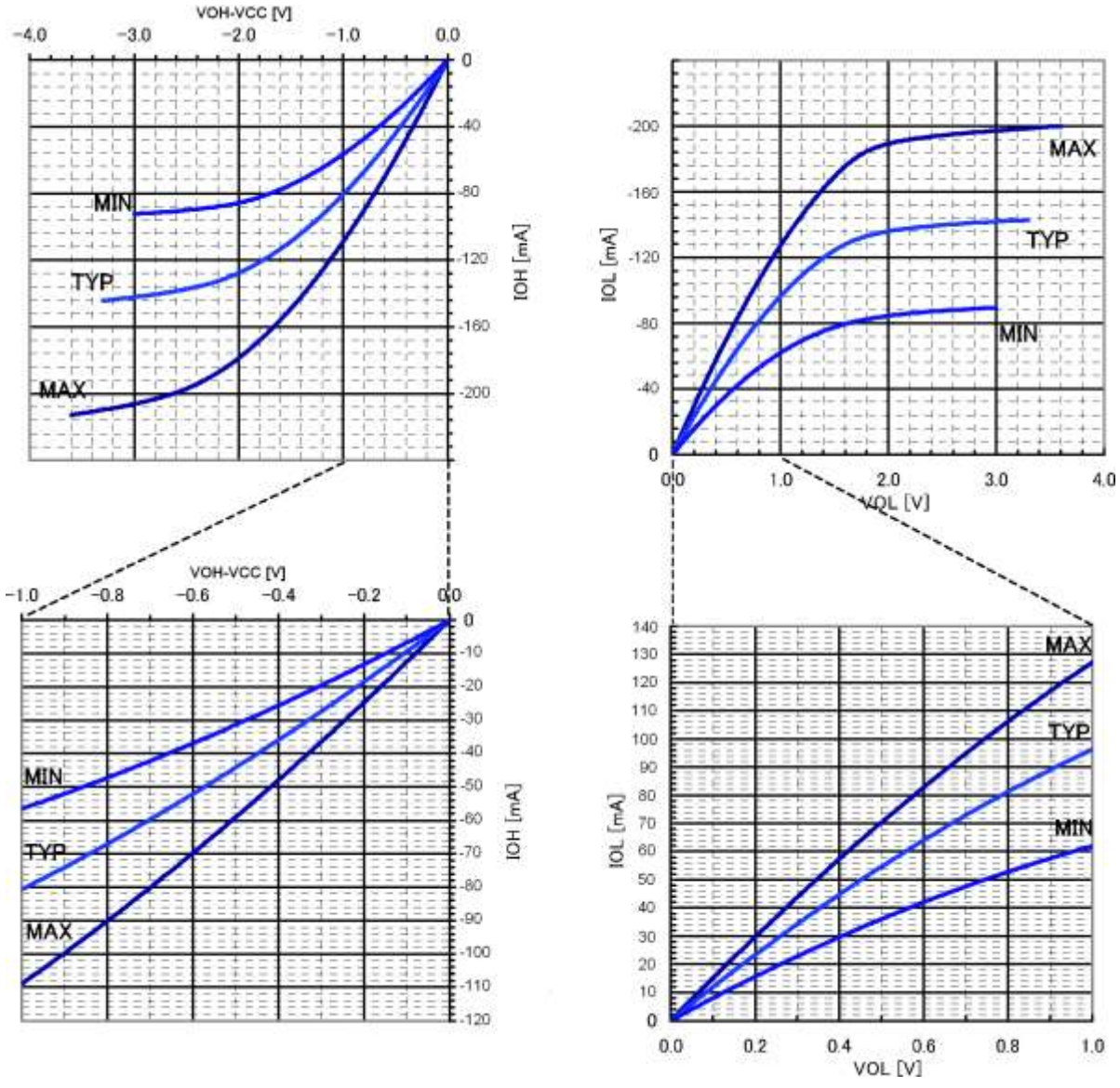


V-I Characteristics 3.3 Vstandard CMOS IO X type

Pins marked in Table 2-1 with Nominal Current I(mA)=±X

Condition:	MIN	Process=	Slow	Tj=	125°C	VCC=	3.0 V
	TYP	Process=Typical		Tj=	25°C	VCC=	3.3 V
	MAX	Process=Fast		Tj=	-40°C	VCC=	3.6 V

Figure 15-3. CMOS IO X type, V-I curves



15.4 Power Consumption

Table 15-4. ATPL210A Power Consumption

Parameter	Condition	Symbol	Rating			Unit
			Min.	Typ.	Max.	
Power Consumption	TA=25°C, VCC=3.3v	P_{25}	--	260	--	mW
Power Consumption (worst case)	TA=85°C, VCC=3.6v	P_{85}	--	--	355	--

15.5 Thermal Data

Table 15-5. ATPL210A Thermal Data

Parameter	Symbol	LQFP144	Unit
Thermal resistance junction-to-ambient steady state	$R_{Theta-ja}$	60(1)	ºC/W
		40(2)	

Notes: 1. Mounted on 2-layer PCB.
2. Mounted on 4-layer PCB.

Theta-ja is calculated based on a standard JEDEC defined environment and is not reliable indicator of a device's thermal performance in a non-JEDEC environment. The customer should always perform their own calculations/simulations to ensure that their system's thermal performance is sufficient.

15.6 Oscilator

Figure 15-4. External Crystal configuration

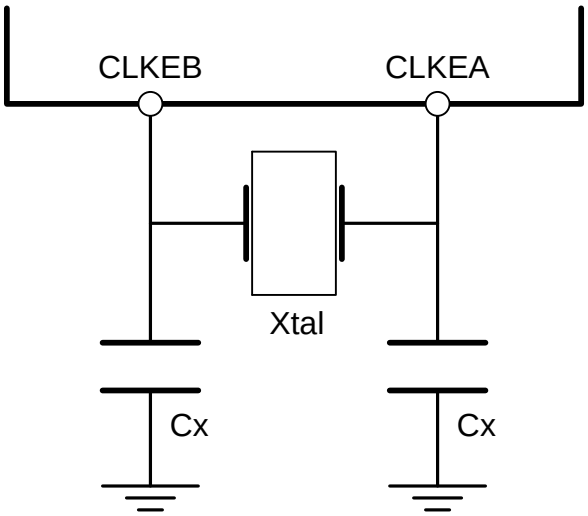
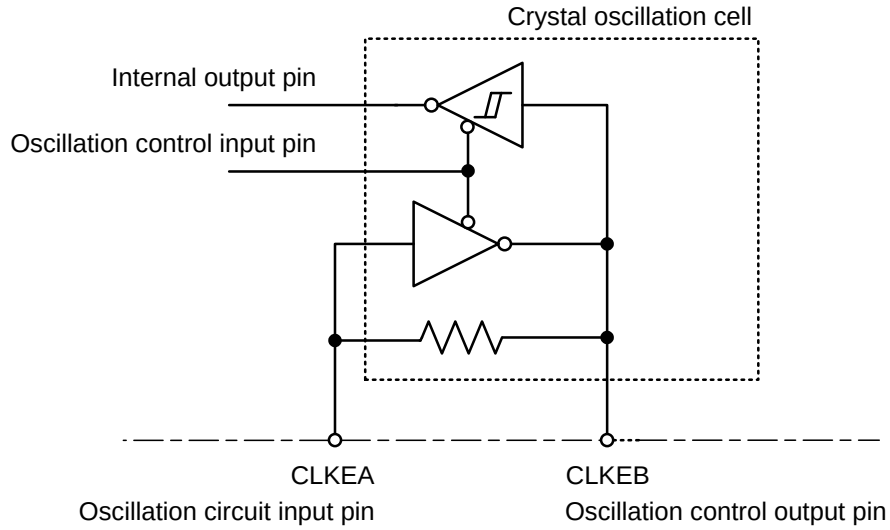


Table 15-6. External oscillator parameters

Parameter	Test Condition	Symbol	Rating			Unit
			Min.	Typ.	Max.	
Crystal Oscillator frequency	fundamental	Xtal		20		MHz
External Oscillator Capacitance	See Figure 15-4	Cx	5	18	30	pF
H-level Input Voltage		XVIH	2	-	VCC+0.3	V
L-level Input Voltage		XVIL	-0.3	-	0.8	
External Oscillator Parallel Resistance		Rp	*not needed*			Ω
External Oscillator Series Resistance		Rs	*not needed*			

Notes:
1. The crystal should be located as close as possible to CLKEB and CLKEA pins.
2. Recommended value for Cx is 18pF. This value may depend on the specific crystal characteristics.
3. Crystal Stability/Tolerance/Ageing values must be selected according to standard PRIME requirements.

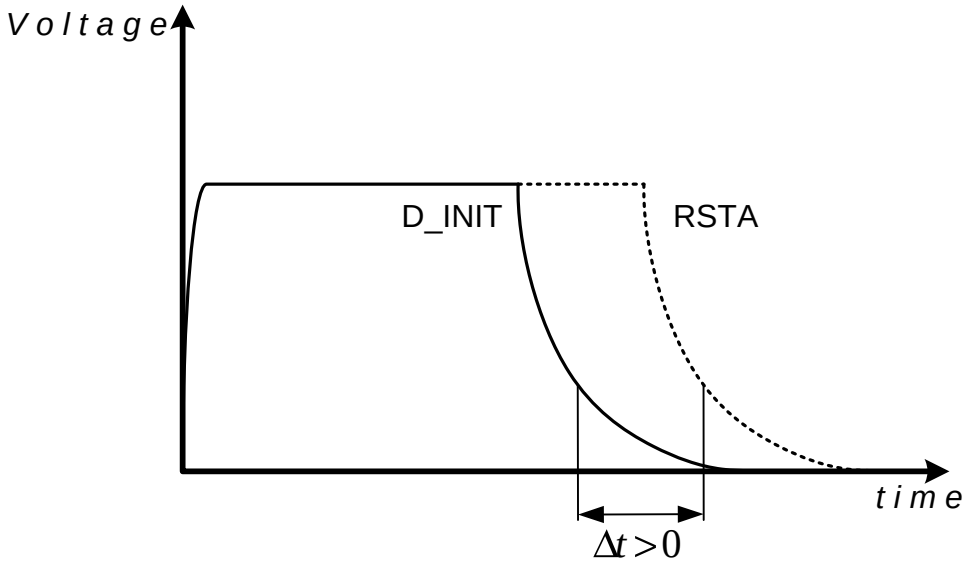
Figure 15-5. Internal Oscillator Cell



15.7 Power On

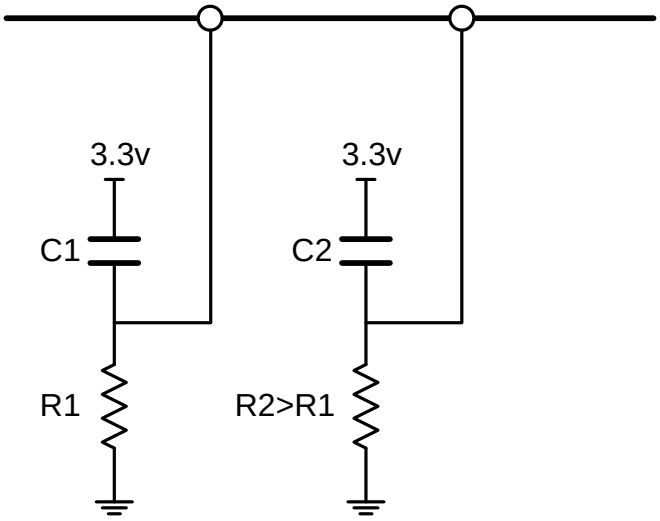
During power-on, D_INIT should be released before asynchronous reset signal RSTA in order to ensure proper system start up.

Figure 15-6. D_INIT & RSTA realease sequence during power-on



Not minimum time is required between both releases, $\Delta t > 0$, so a simple RC circuit is enough to satisfy this requirement.

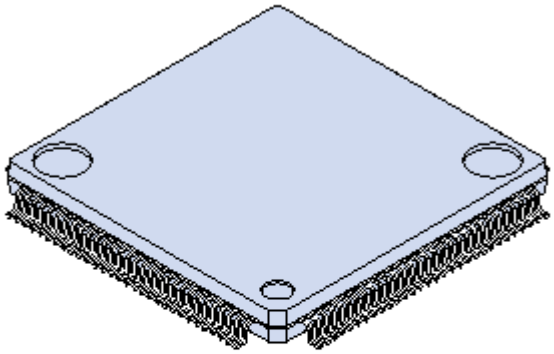
Figure 15-7. Example circuit



16. Mechanical Characteristics

Figure 16-1. 120-lead LQFP Package Mechanical Drawing

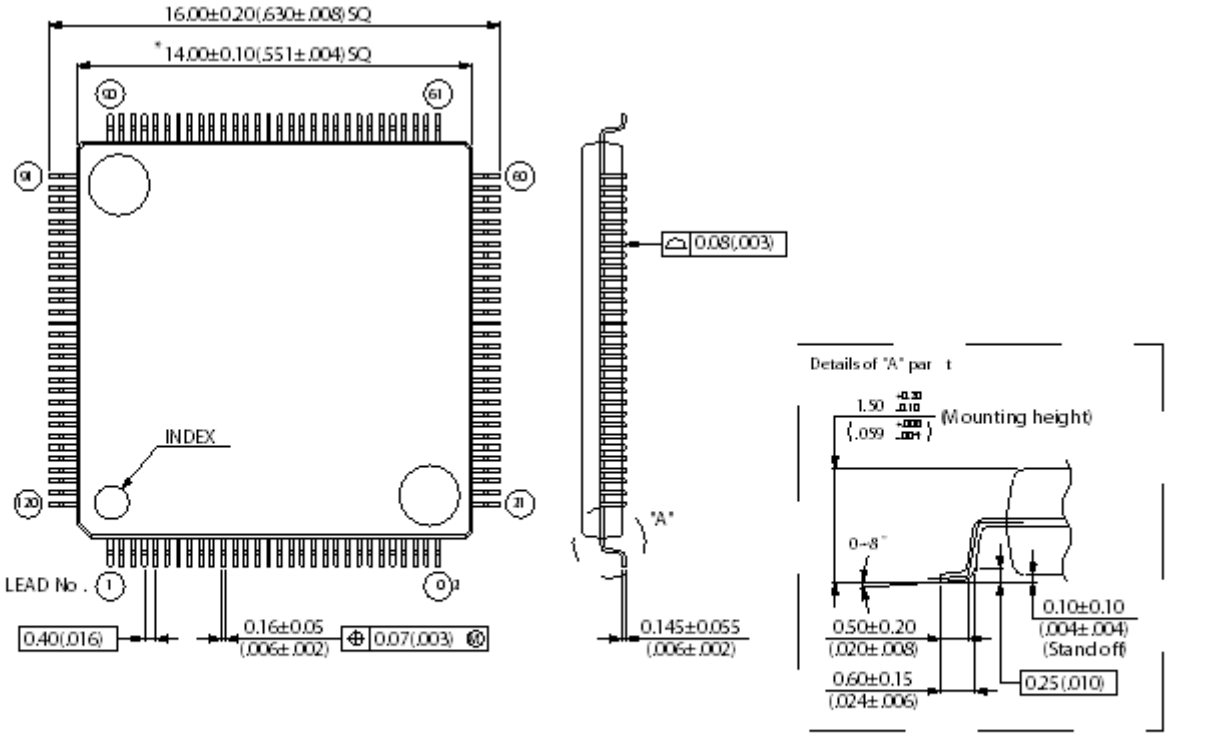
120-pin plastic LQFP



Lead pitch	0.40 mm
Package width · package length	14.0 mm · 14.0 mm
Lead shape	Gullwing
Sealing method	Plastic mold
Mounting height	1.70 mm MAX
Code (Reference)	P-LFQFP120-14 · 14-0.40

120-pin plastic LQFP

Note 1) * : These dimensions do not include resin protrusion.
Note 2) Pins width and pins thickness include plating thickness.
Note 3) Pins width do not include tie bar cutting remainder.



Dimensions in mm (inches).
Note: The values in parentheses are reference values.

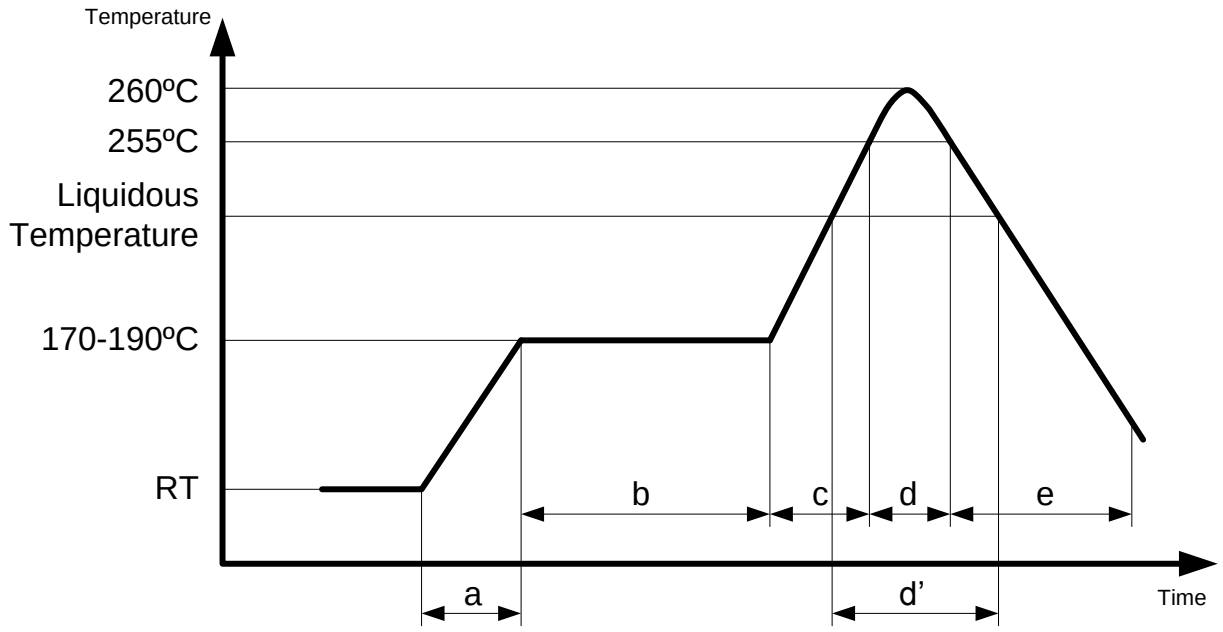
17. Recommended mounting conditions

17.1 Conditions of Standard Reflow

Table 17-1. Conditions of standard Reflow

Items	Contents	
Method	IR(Infrared Reflow)/Convection	
Times	2	
Floor Life	Before unpacking	Please use within 2 years after production
	From unpacking to second reflow	Within 8 days
	In case over period of floor life	Baking with 125ºC +/- 3ºC for 24hrs +2hrs/-0hrs is required. Then please use within 8 days. (please remember baking is up to 2 times)
Floor Life Condition	Between 5ºC and 30ºC and also below 70%RH required. (It is preferred lower humidity in the required temp range.)	

Figure 17-1. Temperature Profile



Note: H rank: 260ºC Max

a: Average ramp-up rate: 1ºC/s to 4ºC/s

b: Preheat & Soak: 170ºC to 190ºC, 60s to 180s

c: Average ramp-up rate: 1ºC/s to 4ºC

d: Peak temperature: 260ºC Max, up to 255ºC within 10s

d': Liquidous temperature: Up to 230ºC within 40s or
Up to 225ºC within 60s or
Up to 220ºC within 80s

e: Cooling: Natural cooling or forced cooling

17.2 Manual Soldering

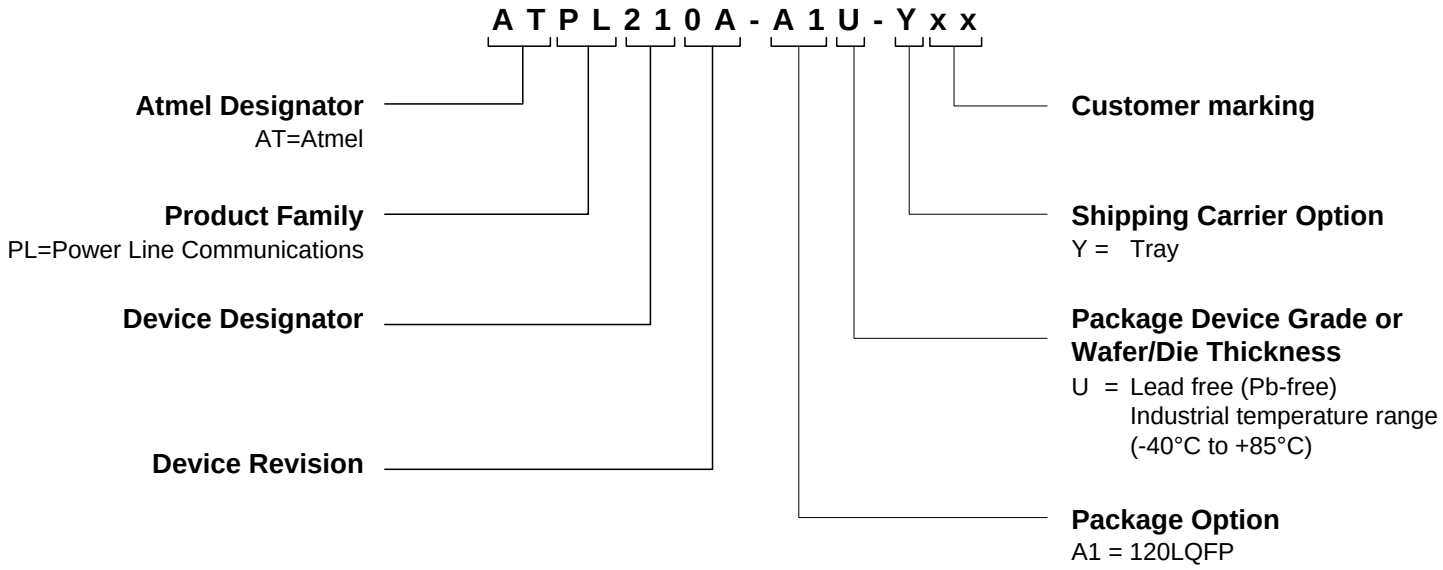
Table 17-2. Conditions of Manual Soldering

Items	Contents	
Floor life	Before unpacking	Please use within 2 years after production
	From unpacking to Manual Soldering	Within 2 years after production (No control required for moisture adsorption because it is partial heating)
Floor life condition	Between 5°C and 30°C and also below 70%RH required. (It is preferred lower humidity in the required temp range.)	
Solder Condition	Temperature of soldering iron: Max 400°C, Time: Within 5 seconds/pin *Be careful for touching package body with iron	

18. Ordering Information

Table 18-1. Atmel ATPL210A Ordering Codes

Atmel Ordering Code	Package	Package Type	Temperature Range
ATPL210A-A1U-Y	120 LQFP	Pb-Free	Industrial (-40ºC to 85º)



19. Revision History

Doc. Rev.	Date	Comments
A	06/18/2012	Initial release

Atmel Corporation
1600 Technology Drive
San Jose, CA 95110
USA
Tel: (+1)(408) 441-0311
Fax: (+1)(408) 487-2600
www.atmel.com

Atmel Asia Limited
Unit 01-5 & 16, 19F
BEA Tower, Millennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
HONG KONG
Tel: (+852) 2245-6100
Fax: (+852) 2722-1369

Atmel Munich GmbH
Business Campus
Parkring 4
D-85748 Garching b. Munich
GERMANY
Tel: (+49) 89-31970-0
Fax: (+49) 89-3194621

Atmel Japan G.K.
16F Shin-Osaki Kangyo Building
1-6-4 Osaki
Shinagawa-ku, Tokyo 141-0032
JAPAN
Tel: (+81)(3) 6417-0300
Fax: (+81)(3) 6417-0370

X-ON Electronics

Largest Supplier of Electrical and Electronic Components

Click to view similar products for Network Controller & Processor ICs ***category:***

Click to view products by Microchip ***manufacturer:***

Other Similar products are found below :

COM20019I3V-HT Z8523L10VEG NCN49597MNG BCM63168UKFEBG TMC2074-NU WAV624A1MC S LN25 WAV614A1MC S LN24 73M2901CE-IM/F COM20020I-DZD-TR COM20020I-DZD KSZ8692PBI KSZ9692PB 73M2901CE-IGV/F MPL360BT-I/Y8X COM20019I-DZD COM20020I3V-DZD-TR COM20022I-HT KSZ8695P LAN9360A-I/CQB-100 LAN9360A-I/CQBT-100 MPL360B-I/SCB MIC3001GML-TR 2751807 NCN49599MNG TMC2072-MT ST7590 73M2901CE-IGVR/F Z8523316ASG Z8523010PEG Z8523008PSG Z8523020VSG Z8523016VEG Z8523010VSG Z8523010VEG Z8523008VSG Z8523016VSG Z8523008VEG Z8523L16VSG AMIS49587C5872G COM20020I-HT CY8CPLC20-28PVXI KSZ8692XPB KSZ8695X ST7580TR