SAMSUNG ELECTRONICS

S5N8947

MCU for DSL

DATA SHEET

Rev 1.8
Oct. 25, 2001

SAMSUNG Electronics Co., LTD.

IMPORTANT NOTICE

SAMSUNG reserves the right to make changes to its products or to discontinue any semiconductor product or service without notice, and advises its customers to obtain the latest version of relevant information to verify, before placing orders, that the information being relied on is current.

CONTENTS

1. GENERAL DESCRIPTION

Samsung's S5N8947 16/32-bit RISC microcontroller is a cost-effective, high-performance microcontroller solution. The S5N8947 is designed as 2-channel 10/100Mbps Ethernet controller for use in managed communication hubs and routers. The S5N8947 also provides ATM Layer SAR (Segmentation and Reassembly) function with UTOPIA interface and the full-rate USB (Universal Serial Bus) function.

The S5N8947 is built around an outstanding CPU core: the 16/32-bit ARM7TDMI RISC processor designed by Advanced RISC Machines, Ltd. The ARM7TDMI core is a low-power, general purpose, microprocessor macro-cell that was developed for use in application-specific and custom-specific integrated circuits. Its simple, elegant, and fully static design is particularly suitable for cost-sensitive and power-sensitive applications.

Important peripheral functions including an UART channel, 2-channel GDMA, three 32-bit timers, watchdog timer, I^2C bus controller, SPI, and programmable I/O ports are supported. Built-in logic including an interrupt controller, DRAM controller, and a controller for ROM/SRAM and flash memory are also supported. The S5N8947's System Manager provides an internal 32-bit system bus arbiter and an external memory controller including control logic for a PCMCIA socket interface.

To reduce total system cost, the S5N8947 offers a unified cache, 2-channel 10/100Mbps Ethernet controller, SAR and USB. Most of the on-chip function blocks have been designed using an HDL synthesizer and the S5N8947 has been fully verified in Samsung's state-of-the-art ASIC test environment.

Item	S5N8946	S5N8947
Architecture	Only one mode	Two modes are supported: Mode 1. MII + UTOPIA + USB Mode 2. 2*MII + UTOPIA + USB
	2 Timer	3 Timer
	-	1 Watchdog Timer
	-	SPI Interface support
	-	PCMCIA support
Function	UTOPIA Level 1 Support	UTOPIA Level 1/2 Support
	Seven Wire Support (10 Mbps Ethernet Support)	MII/Seven Wire Support (10/100 Mbps Ethernet Support)
	USB support Byte access.	USB support Word access and DMA operation.
	SAR support hardwired little Endian.	SAR support hardwired Big/Little Endian.
Performance	50 MHz operation	72 MHz operation
	4K Unified Cache	8K Unified Cache
Operation condition	3.3V	1.8V
Package	240 QFP	208 LQFP

Table 1 S5N8946 vs. S5N8947

2. FEATURES

- ✓ 8-Kbyte unified cache
- ✓ SAR (Segmentation and Reassembly)
- ✓ UTOPIA (the Universal Test & Operations PHY Interface for ATM) Level 1/2 Interface
- ✓ 2-channel 10/100Mbps Ethernet
- ✓ Full-rate USB controller
- ✓ 2-CH GDMA (General Purpose Direct Memory Access)
- ✓ UART (Universal Asynchronous Receiver and Transmtter)
- ✓ 3 programmable 32bits Timers
- ✓ Watchdog Timer
- ✓ 18 Programmable I/O ports
- ✓ Interrupt controller
- ✓ I^2C controller
- ✓ SPI (Serial Peripheral Interface)
- ✓ Built-in PLLs for System/USB
- ✓ PCMCIA ‘memory and I/O’ master modes
- ✓ Cost effective JTAG-based debug solution
- ✓ Boundary scan
- ✓ 3.3V I/Os and 1.8V core supply voltage
- ✓ Operating Frequency Up to 72MHz
- ✓ 208 LQFP Package

3. FUNCTIONAL BLOCK DESCRIPTIONS

3.1. Block Diagram

3.1.1 Mode 1 (1 SAR + 1 MII + 1 USB)

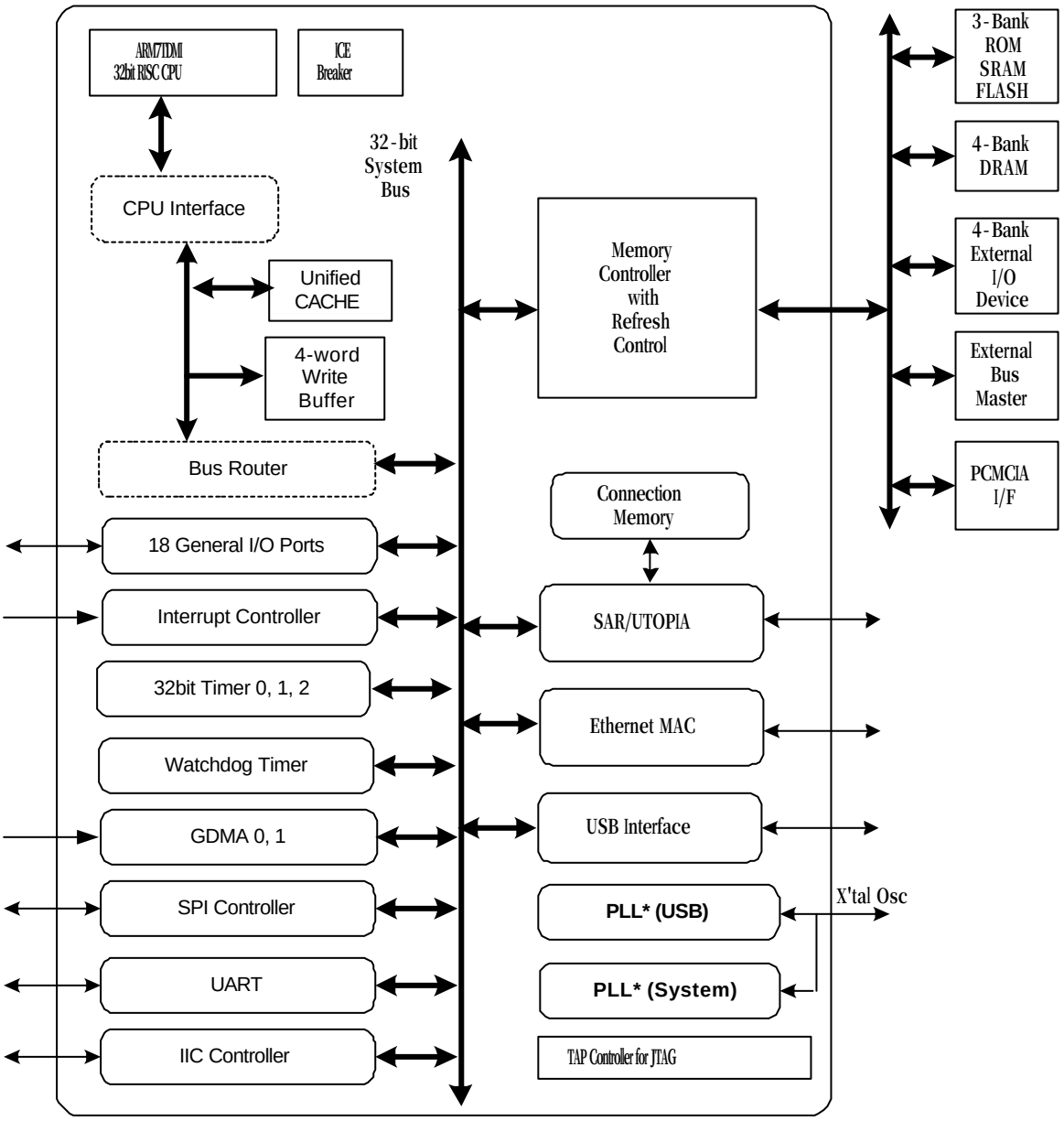

Figure 1 Top Block Diagram: Mode 1

3.1.2 Mode 2 (1 SAR + 2 MII + 1 USB)

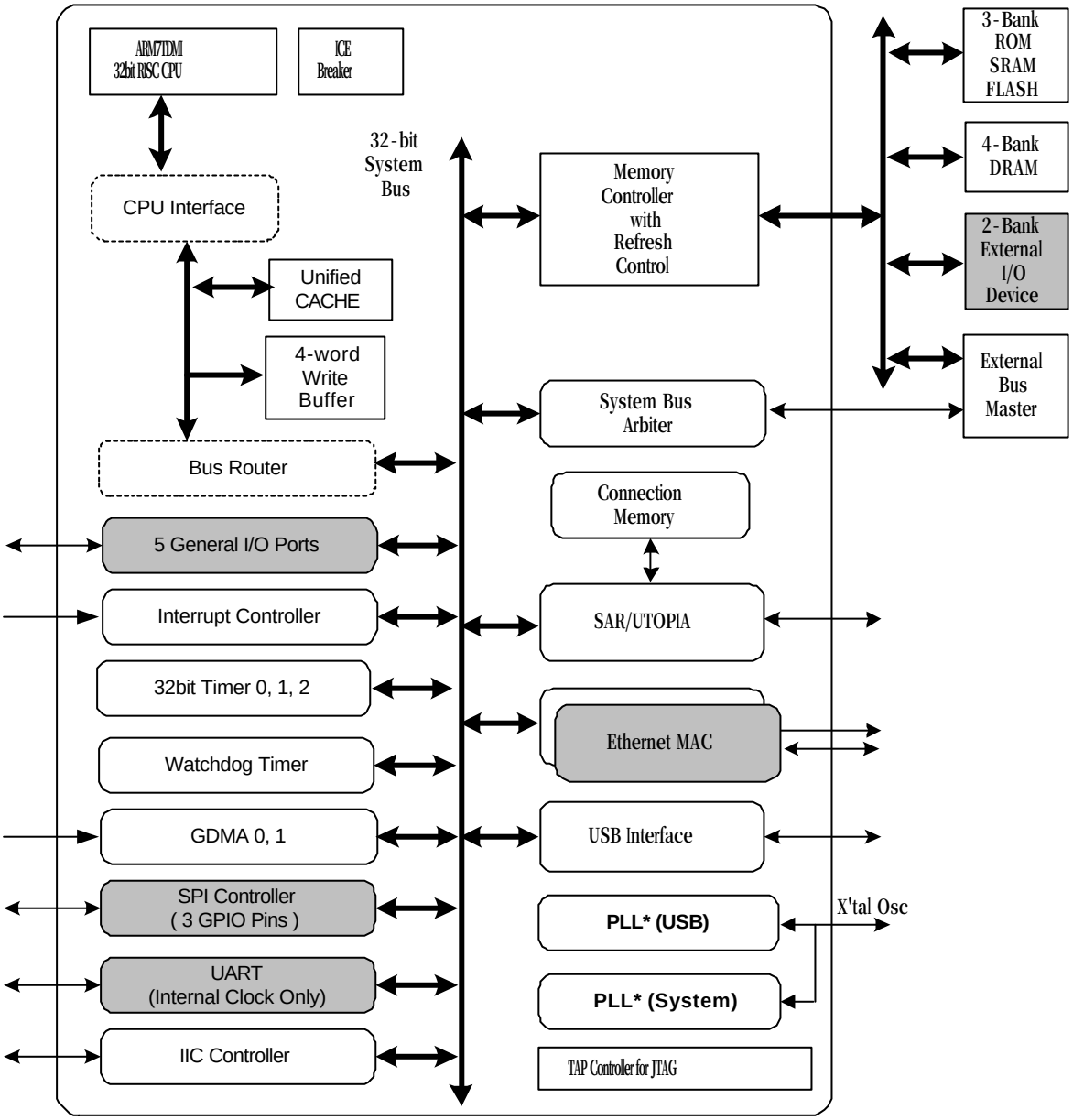

Figure 2 Top Block Diagram: Mode 2

3.2. Architecture

Integrated system for embedded Ethernet / USB / SAR
Fully 16/32-bit RISC architecture
Efficient and powerful ARM7TDMI core
Little/Big-Endian mode is fully supported. (The internal register supports word access only.)
Cost-effective JTAG-based debug solution
Supports Boundary Scan

3.3. System Manager

8/16/32-bit external bus support for ROM/SRAM, flash memory, DRAM and external I/O
One external bus master with bus request/acknowledge pins
Supports EDO/normal or SDRAM
Programmable access cycle
Four-word depth write buffer
Cost-effective memory-to-peripheral DMA interface
Supports PCMCIA 'memory and I/O' master mode

3.4. Unified Instruction/Data Cache

Two-way set-associative unified cache (8Kbytes)
Supports LRU (least recently used) Protocol

3.5. SAR/UTOPIA Interface

Directly supports ATM Adaptation Layer Five (AAL5) Segmentation And Reassembly
Segments and reassembles data up to 70Mbps
A glueless UTOPIA level 1/2 interface is supprted (for receiving and transmitting ATM cells with SAR, it is a standard ATM interface between ATM link and physical layer).

3.6. ETHERNET

2-Channel 10/100Mbps Ethernet Controller
4 DMA engines with burst mode
Full compliance with IEEE standard 802.3
Supports MII interface (7-wire 10-Mbps interface is also supported).

3.7. USB Controller

Supports 12Mbps full rate function for universal serial bus

3.8. DMA Controller

2-channel general purpose DMA (for memory-to-memory, memory-to-SPI, SPI-to-memory, UART-to-memory, memory-to-UART data transfers without CPU intervention)
Initiated by a software or a external DMA request
Increments or decrements source or destination address in 8-bit, 16-bit or 32-bit data transfers

3.9. UART

UART block with DMA-based or interrupt-based operation
Supports 5-bit, 6-bit, 7-bit, or 8-bit serial data transmit and receive
Programmable baud rates
Infra-red (IR) TX/RX support (IrDA)

3.10. Timers

Three programmable 32-bit timers
Interval mode or toggle mode operation
Supports a watchdog timer

3.11. Programmable I/O

18 programmable I/O ports
Pins individually configurable to input, output, or I/O mode for dedicated signals

3.12. Interrupt Controller

23 interrupt sources, including 7 external interrupt sources
Normal or fast interrupt mode (IRQ, FIQ)
Prioritized interrupt handling

3.13. I²C Serial Interface

Single master mode operation only

3.14. SPI

Full duplex operation
Work with data characters from 4 to 32 bits long
Supports GDMA mode for SPI transmission and reception
Single master SPI modes only supported
Programmable baud rate generator
Programmable clock phase and polarity

3.15. PLLs

The external clock can be multiplied by on-chip PLLs to provide high frequency System/USB clock
The input frequency is fixed to 12 MHz
The output frequency is 6 times the input clock for System
The output frequency is 4 times the input clock for USB

4. PIN DESCRIPTIONS

4.1. Pin Configuration

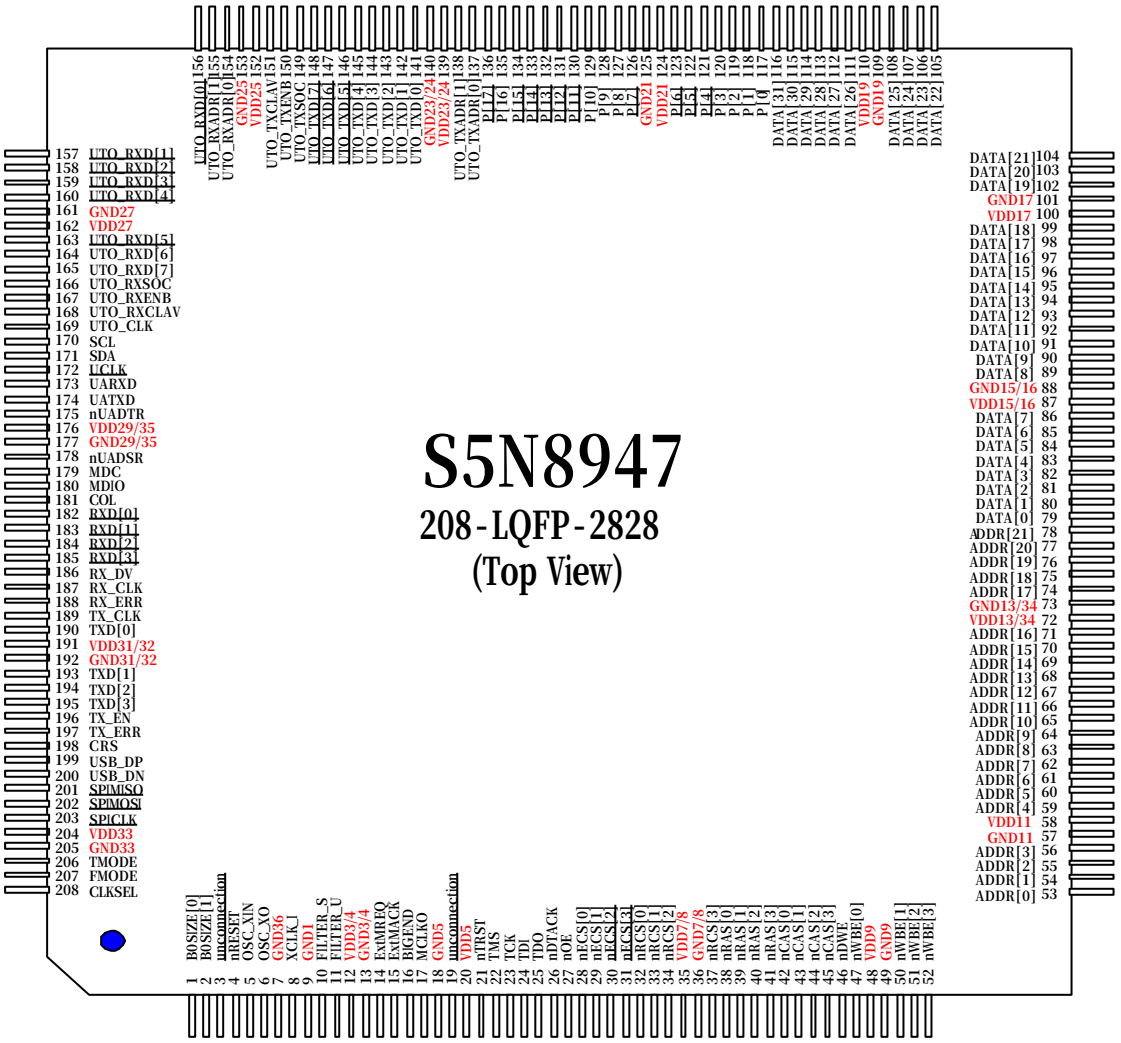

Figure 3 S5N8947 Pin Configuration

NOTES

✓ Under-bar in the Figure 2 means the muxing pins.

4.2. Logic Symbol Diagram

4.2.1 Mode 1 (1 SAR + 1 MII + 1 USB)

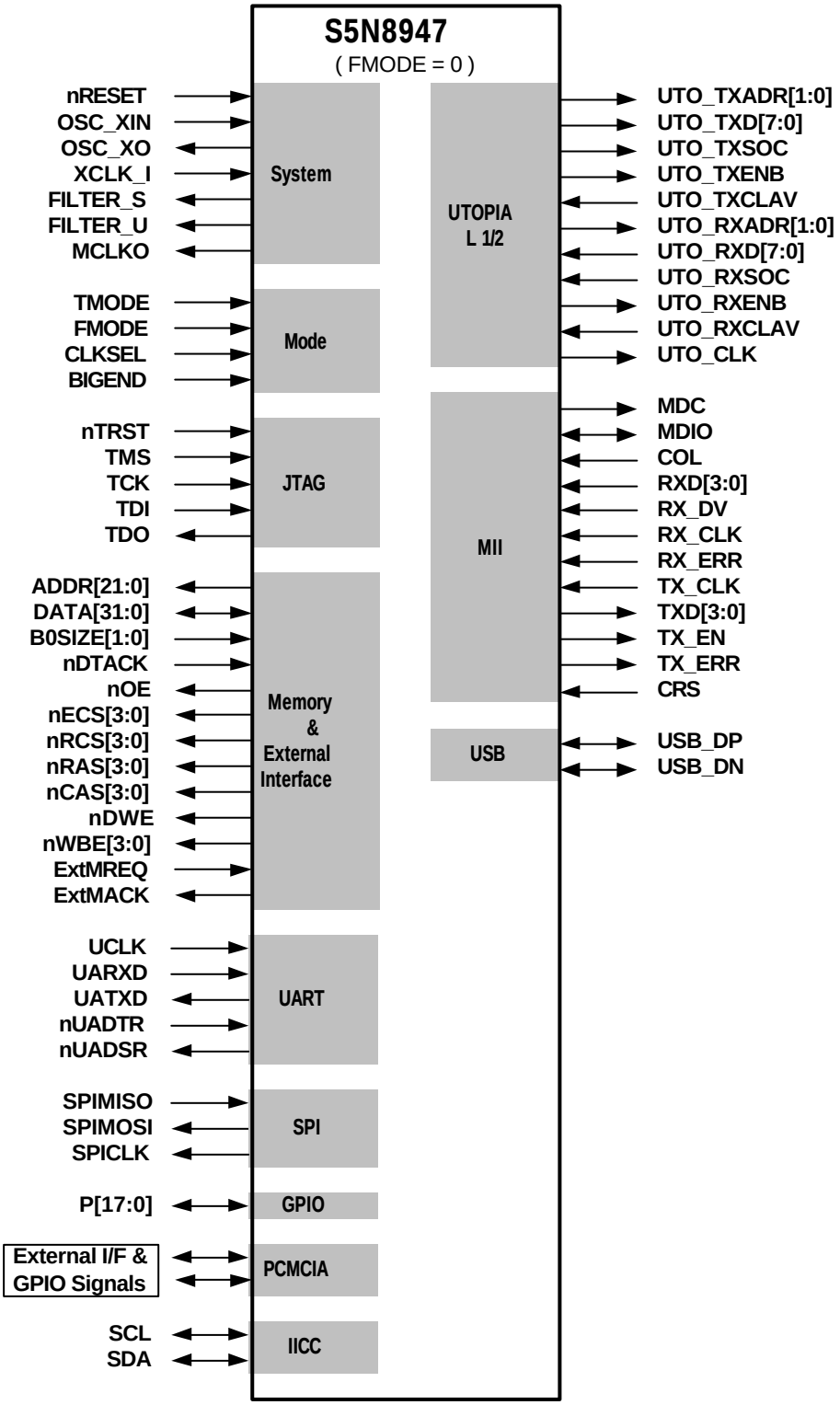

Figure 4 S5N8947 Logic Symbol Diagram (Mode 1)

4.2.2 Mode 2 (1 SAR + 2 MII + 1 USB)

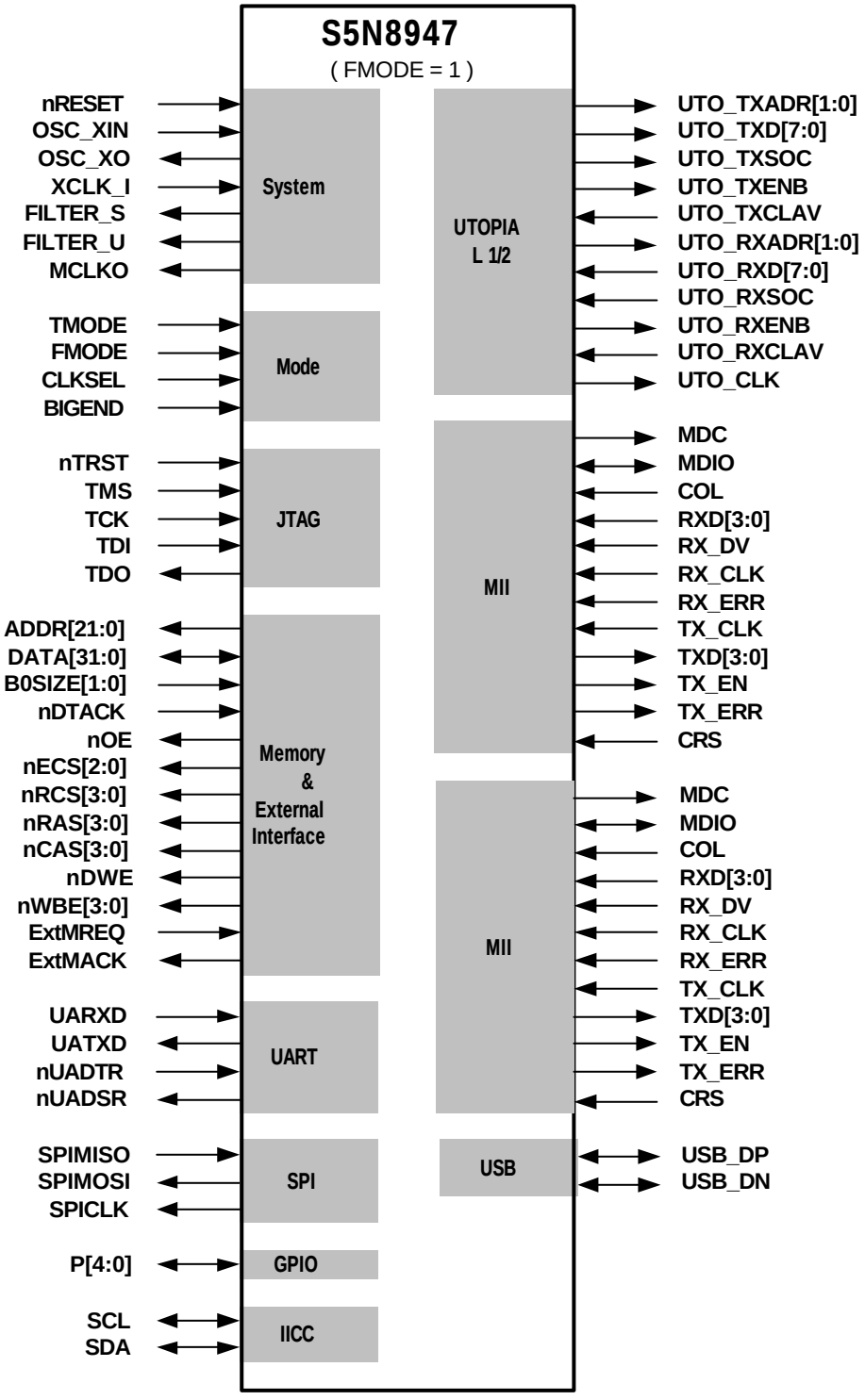

Figure 5 S5N8947 Logic Symbol Diagram (Mode 2)

4.3 Pin Descriptions with the Pin number and Pad type

Pin No	Pin Name	I/O Type	Pad type	Descriptions
1	B0SIZE[0]	I	Phic	
2	B0SIZE[1]	I	Phic	
*3	unconnection	I	Phic	Muxing with sRX_CLK
4	nRESET	I	Phtis	
5	OSC_XIN	I	Phsoscm2	
6	OSC_XO	O	Phsoscm2	
7	GND36	P	Vss3o	
8	XCLK_I	I	phic	
9	GND1	P	Vbb1_abb	
10	FILTER_S	O	Poar50_abb	
11	FILTER_U	O	Poar50_abb	
12	VDD3/4	P	Vdd1t_abb	1.8V
13	GND3/4	P	Vss1t_abb	
14	ExtMREQ	I	Phic	
15	ExtMACK	O	Phob1	
16	BIGEND	I	Phicd	
17	MCLKO	O	Phob4	
18	GND5	P	Vss3p	
*19	Unconnection	B	Phbcut4	Muxing with sMDIO
20	VDD5	P	Vdd3p	3.3V
21	nTRST	I	Phicu	
22	TMS	I	Phicu	
23	TCK	I	phic	
24	TDI	I	Phicu	
25	TDO	O	Phtot2	
26	NDACK	I	Phicu	
27	NOE	O	Phot4	
28	NECS[0]	O	Phot4	
29	NECS[1]	O	Phot4	
*30	NECS[2]	O	Phot4	Muxing with sTXD[3]
*31	NECS[3]	O	Phot4	Muxing with sMDC
32	nRCS[0]	O	Phot4	
33	nRCS[1]	O	Phot4	
34	nRCS[2]	O	Phot4	
35	VDD7/8	P	Vdd3o	3.3V
36	GND7/8	P	Vss3o	
37	nRCS[3]	O	Phot4	Not PCMCIA select
38	nRAS[0]	O	Phot4	
39	nRAS[1]	O	Phot4	
40	nRAS[2]	O	Phot4	
41	nRAS[3]	O	Phot4	
42	nCAS[0]	O	Phot4	
43	nCAS[1]	O	Phot4	
44	nCAS[2]	O	Phot4	
45	nCAS[3]	O	Phot4	
46	nDWE	O	Phot4	

47	nWBE[0]	O	Phot4	
48	VDD9	P	Vdd1ih	1.8V
49	GND9	P	Vss3i	
50	nWBE[1]	O	Phot4	nWBE[1]/IORD(PCMCIA only)
51	nWBE[2]	O	Phot4	nWBE[2]/IOWR(PCMCIA only)
52	nWBE[3]	O	Phot4	
53-56	ADDR[0:3]	O	Phot4	
57	GND11	P	Vss3i	
58	VDD11	P	Vdd1ih	1.8V
59-71	ADDR[4:16]	O	Phot4	
72	VDD13/34	P	Vdd3o	3.3V
73	GND13/34	P	Vss3o	
74-78	ADDR[17:21]	O	Phot4	
79-86	DATA[0:7]	B	Phbcut4	
87	VDD15/16	P	Vdd3o	3.3V
88	GND15/16	P	Vss3o	
89-99	DATA[8:18]	B	Phbcut4	
100	VDD17	P	Vdd1ih	1.8V
101	GND17	P	Vss3i	
102-108	DATA[19-25]	B	Phbcut4	
109	GND19	P	Vss3I	
110	VDD19	P	Vdd1ih	1.8V
111-116	DATA[26-31]	B	Phbcut4	
117-120	P[0:3]	B	Phbcut4	
*121	P[4]	B	Phbcut4	Muxing with sTXD[0]
*122	P[5]	B	Phbcut4	Muxing with sTXD[1]
*123	P[6]	B	Phbcut4	Muxing with sTXD[2]
124	VDD21	P	Vdd3p	3.3V
125	GND21	P	Vss3p	
*126	P[7]	B	Phbcut4	Muxing with sRX_DV
127-129	P[8-10]	B	Phbcut4	
*130	P[11]	B	Phbcut4	Muxing with sCRS
*131	P[12]	B	Phbcut4	Muxing with sRXD[0]
*132	P[13]	B	Phbcut4	Muxing with sRXD[1]
*133	P[14]	B	Phbcut4	Muxing with sRXD[2]
*134	P[15]	B	Phbcut4	Muxing with sRXD[3]
135	P[16]	B	Phbcut4	
*136	P[17]	B	Phbcut4	Muxing with sRX_ERR
137-138	UTO_TXADR[0:1]	O	Phob4	
139	VDD23/24	P	Vdd3o	3.3V
140	GND23/24	P	Vss3o	
141-145	UTO_TXD[0:4]	O	Phob4	
*146	UTO_TXD[5]	O	Phob4	Muxing with bist_errob
*147	UTO_TXD[6]	O	Phob4	Muxing with bist_diag
*148	UTO_TXD[7]	O	Phob4	Muxing with bist_done
149	UTO_TXSOC	O	Phob4	
150	UTO_TXENB	O	Phob4	
151	UTO_TXCLAV	I	Phtis	
152	VDD25	P	Vdd1ih	1.8V

153	GND25	P	Vss3i	
154-155	UTOP_RXADR[0:1]	O	Phob4	
*156	UTO_RXD[0]	I	phtis	Muxing with bist_on
*157	UTO_RXD[1]	I	phtis	Muxing with bist_mode
*158	UTO_RXD[2]	I	phtis	Muxing with bist_memsel[0]
*159	UTO_RXD[3]	I	phtis	Muxing with bist_memsel[1]
*160	UTO_RXD[4]	I	phtis	Muxing with bist_memsel[2]
161	GND27	P	Vss3i	
162	VDD27	P	Vdd1ih	1.8V
*163	UTO_RXD[5]	I	phtis	Muxing with bist_memsel[3]
164-165	UTO_RXD[6:7]	I	phtis	
166	UTO_RXSOC	I	Phtis	
167	UTO_RXENB	O	Phob4	
168	UTO_RXCLAV	I	Phtis	
169	UTO_CLK	O	Phob4	
170	SCL	B	Phbcud4	
171	SDA	B	Phbcud4	
*172	UCLK	I	phic	Muxing with sTX_CLK
173	UARXD	I	Phic	
174	UATXD	O	Phob4	
175	nUADTR	I	Phic	
176	VDD29/35	P	Vdd3o	3.3V
177	GND29/35	P	Vss3o	
178	nUADSR	O	Phob4	
179	MDC	O	Phob4	
180	MDIO	B	Phbcut4	
181	COL	I	Phic	7-wire pin
*182	RXD[0]	I	Phic	Muxing with test_mode[0], 7-wire pin
*183	RXD[1]	I	Phic	Muxing with test_mode[1]
*184	RXD[2]	I	Phic	Muxing with test_mode[2]
*185	RXD[3]	I	Phic	Muxing with test_mode[3]
186	RX_DV	I	Phic	
187	RX_CLK	I	Phic	7-wire pin
188	RX_ERR	I	Phic	
189	TX_CLK	I	Phic	7-wire pin
190	TXD[0]	O	Phob4	7-wire pin
191	VDD31/32	P	Vdd3o	3.3V
192	GND31/32	P	Vss3o	
193-195	TXD[1:3]	O	Phob4	
196	TX_EN	O	Phob4	7-wire pin
197	TX_ERR	O	Phob4	
198	CRS	I	Phic	7-wire pin
199	USB_DP	B	Pbusbfs	
200	USB_DN	B	Pbusbfs	
*201	SPIMISO	I	Phic	Muxing with sCOL
*202	SPIMOSI	O	Phob4	Muxing with sTX_EN
*203	SPICLK	O	Phob4	Muxing with sTX_ERR
204	VDD33	P	Vdd1ih	**1.8V**
205	GND33	P	Vss3i	

206	TMODE	I	Phic	
207	FMODE	I	Phic	
208	CLKSEL	I	phic	

4.4 PAD Descriptions

4.4.1 Input PADs

Pad Types	Descriptions
PHIC / PHICS / PHICU	3.3V interface LVCMOS Level Input Buffer
PHIS / PHISD / PHISU	3.3V interface LVCMOS schmitt trigger level input buffer
PHTIS / PHTISD / PHTISU	5V tolerant for 3.3V interface CMOS schmit trigger level input buffer

4.4.2 Output PADs

Pad Types	Descriptions
PHOB (1/4/8)	3.3V LVCMOS Normal Output Buffers
PHOT (1/4/8)	3.3V LVCMOS Tri-State Output Buffers

4.4.3 Bi-direction PADs

Pad Types	Descriptions
PHBCUT4 (PHBaTyz)	3.3V Tri-State Bi-Direction Buffers
PHBCUD4 (PHBaUDyz)	3.3V Open-Drain Bi-Directional Buffers with Pull-Up

4.4.4 Power Pads

Pad Characteristics	Pad Types	Supply Voltage	Descriptions
1.8V Interface Digital I/O	vdd1i	1.8v	1.8V Internal
3.3V Interface Digital I/O	Vdd3p	3.3v	3.3V Pre-Driver
	Vdd3o	3.3v	3.3V Output-Driver
1.8v Interface Digital I/O	Vss1i		Internal GND for 1.8V interface I/O
3.3V Interface Digital I/O	Vss3p		Pre-Driver GND for 3.3v interface I/O
	Vss3o		Output-Driver GND for 3.3v interface I/O
1.8v Interface Analog I/O	Vdd1t_abb	1.8v	1.8v total
1.8v interface analog I/O	Vss1t_abb		Total GND for 1.8v interface I/O
	Vss1_abb		Bulk-Bias GND for 1.8v interface I/O

5. OPERATION DESCRIPTION

5.1. CPU Core Overview

The S5N8947 CPU core is the ARM7TDMI processor, a general purpose, 32-bit microprocessor developed by Advanced RISC Machines, Ltd. (ARM). The core's architecture is based on Reduced Instruction Set Computer (RISC) principles. The RISC architecture makes the instruction set and its related decoding mechanisms simpler and more efficient than those with microprogrammed Complex Instruction Set Computer (CISC) systems. The resulting benefit is high instruction throughput and impressive real-time interrupt response. Pipelining is also employed so that all components of the processing and memory systems can operate continuously. The ARM7TDMI has a 32-bit address bus.

An important feature of the ARM7TDMI processor, and one which differentiates it from the ARM7 processor, is a unique architectural strategy called THUMB. The THUMB strategy is an extension of the basic ARM architecture and consists of 36 instruction formats. These formats are based on the standard 32-bit ARM instruction set, but have been re-coded using 16-bit wide opcodes.

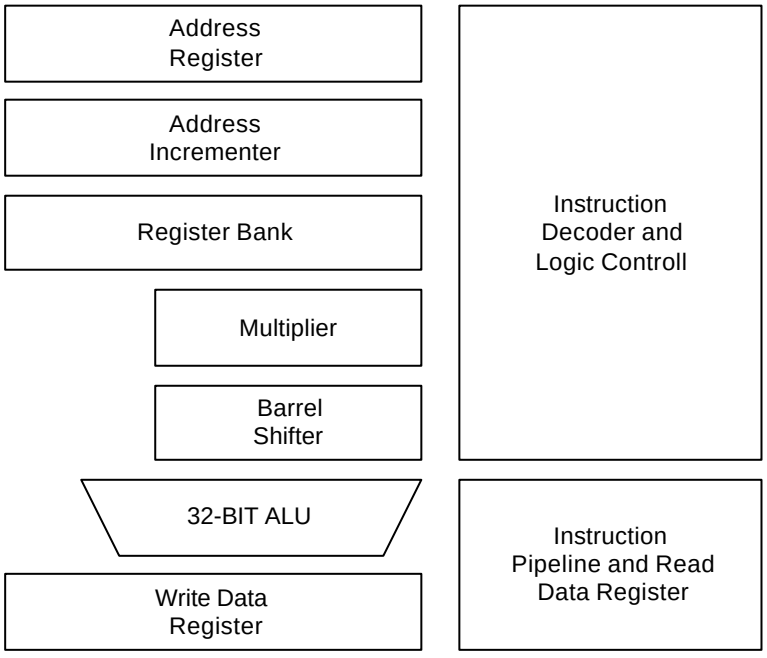

Figure 6 ARM7TDMI Core Block Diagram

Because THUMB instructions are one-half the bit width of normal ARM instructions, they produce very high-density code. When a THUMB instruction is executed, its 16-bit opcode is decoded by the processor into its equivalent instruction in the standard ARM instruction set. The ARM core then processes the 16-bit instruction as it would a normal 32-bit instruction. In other words, the THUMB architecture gives 16-bit systems a way to access the 32-bit performance of the ARM core without incurring the full overhead of 32-bit processing. Because the ARM7TDMI core can execute both standard 32-bit ARM instructions and 16-bit THUMB instructions, it lets you mix routines of THUMB instructions and ARM code in the same address space. In this way, you can adjust code size and performance, routine by routine, to find the best programming solution for a specific application.

5.2. Instruction Set

The S5N8947 instruction set is divided into two subsets: a standard 32-bit ARM instruction set and a *16-bit THUMB instruction set*.

The 32-bit ARM instruction set is comprised of thirteen basic instruction types which can be divided into four broad classes:

- Four types of branch instructions which control program execution flow, instruction privilege levels, and switching between ARM code and THUMB code.
- Three types of data processing instructions which use the on-chip ALU, barrel shifter, and multiplier to perform high-speed data operations in a bank of 31 registers (all with 32-bit register widths).
- Three types of load and store instructions which control data transfer between memory locations and the registers. One type is optimized for flexible addressing, another for rapid context switching, and the third for swapping data.
- Three types of co-processor instructions which are dedicated to controlling external co-processors. These instructions extend the off-chip functionality of the instruction set in an open and uniform way.

NOTES: All 32-bit ARM instructions can be executed conditionally.

The 16-bit THUMB instruction set contains 36 instruction formats drawn from the standard 32-bit ARM instruction set. The THUMB instructions can be divided into four functional groups:

- Four branch instructions.
- Twelve data processing instructions, which are a subset of the standard ARM data processing instructions.
- Eight load and store register instructions.
- Four load and store multiple instructions.

NOTES: Each 16-bit THUMB instruction has a corresponding 32-bit ARM instruction with the identical processing model.

The 32-bit ARM instruction set and the 16-bit THUMB instruction sets are good targets for compilers of many different high-level languages. When assembly code is required for critical code segments, the ARM programming technique is straightforward, unlike that of some RISC processors which depend on sophisticated compiler technology to manage complicated instruction interdependencies.

Pipelining is employed so that all parts of the processor and memory systems can operate continuously. Typically, while one instruction is being executed, its successor is being decoded, and a third instruction is being fetched from memory.

5.3. Operating States

From a programmer's point of view, the ARM7TDMI core is always in one of two operating states. These states, which can be switched by software or by exception processing, are:

- *ARM state* (when executing 32-bit, word-aligned, ARM instructions), and
- *THUMB state* (when executing 16-bit, half-word aligned THUMB instructions).

5.4. Operating Modes

The ARM7TDMI core supports seven operating modes:

- User mode: the normal program execution state
- FIQ (Fast Interrupt Request) mode: for supporting a specific data transfer or channel process
- IRQ (Interrupt ReQuest) mode: for general purpose interrupt handling
- Supervisor mode: a protected mode for the operating system
- Abort mode: entered when a data or instruction pre-fetch is aborted
- System mode: a privileged user mode for the operating system
- Undefined mode: entered when an undefined instruction is executed

Operating mode changes can be controlled by software, or they can be caused by external interrupts or exception processing. Most application programs execute in User mode. Privileged modes (that is, all modes other than User mode) are entered to service interrupts or exceptions, or to access protected resources.

5.5. Registers

The S5N8947 CPU core has a total of 37 registers: 31 general-purpose 32-bit registers, and 6 status registers. Not all of these registers are always available. Which registers are available to the programmer at any given time depends on the current processor operating state and mode.

NOTES: When the S5N8947 is operating in ARM state, 16 general registers and one or two status registers can be accessed at any time. In privileged mode, mode-specific banked registers are switched in.

Two register sets, or banks, can also be accessed, depending on the core's current state: the ARM state register set and the *THUMB state register* set:

- The ARM state register set contains 16 directly accessible registers: R0-R15. All of these registers, except for R15, are for general-purpose use, and can hold either data or address values. An additional (seventeenth) register, the CPSR (Current Program Status Register), is used to store status information.
- The THUMB state register set is a subset of the ARM state set. You can access eight general registers, R0-R7, as well as the program counter (PC), a stack pointer register (SP), a link register (LR), and the CPSR. Each privileged mode has a corresponding banked stack pointer, link register, and saved process status register (SPSR).

The THUMB state registers are related to the ARM state registers as follows:

- THUMB state R0-R7 registers and ARM state R0-R7 registers are identical
- THUMB state CPSR and SPSRs and ARM state CPSR and SPSRs are identical
- THUMB state SP, LR, and PC map directly to ARM state registers R13, R14, and R15, respectively

In THUMB state, registers R8-R15 are not part of the standard register set. However, you can access them for assembly language programming and use them for fast temporary storage, if necessary.

5.6. Exceptions

An exception arises whenever the normal flow of program execution is interrupted. For example, when processing must be diverted to handle an interrupt from a peripheral. The processor's state just prior to handling the exception must be preserved so that the program flow can be resumed when the exception routine is completed. Multiple exceptions may arise simultaneously.

To process exceptions, the S5N8947 uses the banked core registers to save the current state. The old PC value and the CPSR contents are copied into the appropriate R14 (LR) and SPSR register. The PC and mode bits in the CPSR are forced to a value which corresponds to the type of exception being processed.

The S5N8947 core supports seven types of exceptions. Each exception has a fixed priority and a corresponding privileged processor mode, as shown in following Table

Exception	Mode on Entry	Priority
Reset	Supervisor mode	1 (highest)
Data abort	Abort mode	2
FIQ	FIQ mode	3
IRQ	IRQ mode	4
Prefetch abort	Abort mode	5
Undefined instruction	Undefined mode	6
SWI	Supervisor mode	6 (lowest)

Table 2 S5N8947 CPU Exceptions

6. HARDWARE STRUCTURE

6.1. System Manager

6.1.3. Overview

The S5N8947 microcontroller's System Manager has the following functions.

- Arbitrates system bus access requests from several master blocks, based on fixed priorities.
- Provides the required memory control signals for external memory accesses. For example, if a master block such as the DMA controller or the CPU generates an address, which corresponds to a DRAM bank, the System Manager's DRAM controller generates the required normal/EDO or SDRAM access signals. The interface signals for normal/EDO or SDRAM can be switched by SYSCFG[31].
- Provides the required signals for bus traffic between the S5N8947 and ROM/SRAM and the external I/O banks.
- Compensates for differences in bus width for data transfer between the external memory bus and the internal data bus.
- Supports both little and big endian for external memory or I/O devices. Internal registers, however, operate under big-endian mode.
- Supports both motorola mode and intel mode for external I/O devices
- Supports an external bus master with bus request(ExtMREQ) and bus acknowledge(ExtMACK)
- Supports PCMCIA 'memory and I/O' master mode

6.1.4. System Manager Registers

To control external memory operations, the System Manager uses a dedicated set of special registers. By programming the values in the System Manager special registers, you can specify such things as:

- Memory type
- External data access cycle
- External memory and I/O device access cycle
- Memory bank locations
- Size of each memory bank to be used for arbitrary address spacing

The System Manager uses special register setting to control the generation and processing of the control signals, addresses, and data that are required by external devices in a standard system configuration. Special registers are also used to control access to ROM/SRAM/Flash banks, a PCMCIA interface, up to four DRAM banks and four external I/O banks, and a special register mapping area.

The address resolution for each memory bank base pointer is 64 Kbytes (16 bits). The base address pointer is 10 bits. This gives a total addressable memory bank space of 16 M words.

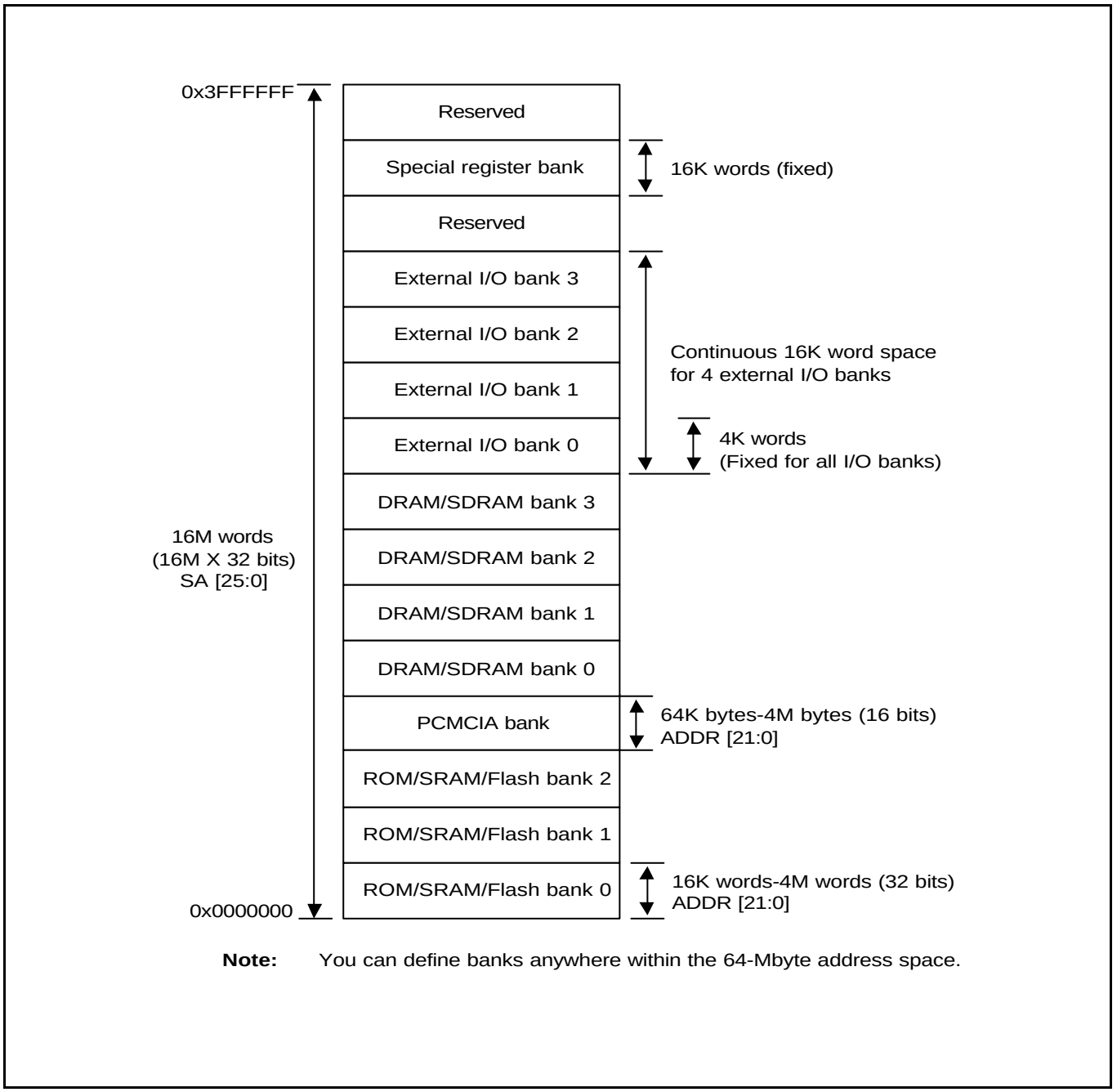

Figure 7 S5N8947 System Memory Map

6.1.5. System Memory Map

Followings are several important features to note about the S5N8947 system memory map:

- The size and location of each memory bank is determined by the register settings for "current bank base pointer" and "current bank end pointer". You can use this base/next bank pointer concept to set up a consecutive memory map. To do this, you set the base pointer of the "next bank" to the same address as the next pointer of the "current bank". Please note that when setting the bank control registers, the address boundaries of consecutive banks must not overlap. This can be applied even if one or more banks are disabled.
- Four external I/O banks are defined in a continuous address space. A programmer can only set the base pointer for external I/O bank 0. The start address of external I/O bank 1 is then calculated as the external I/O bank 0 start address +16 K. Similarly, the start address for external I/O bank 2 is the external I/O bank 0 start address + 32 K, and the start address for external I/O bank 3 is the external I/O bank 0 start address + 48 K. Therefore, the total consecutive addressable space of the four external banks is defined as the start address of external I/O bank 0 + 64 K bytes.
- Within the addressable space, the start address of each I/O bank is not fixed. You can use bank control registers to assign a specific bank start address by setting the bank's base pointer. The address resolution is 64 K bytes. The bank's start address is defined as "base pointer << 16" and the bank's end address (except for external I/O banks) is "next pointer << 16 – 1".

After a power-on or system reset, all bank address pointer registers are initialized to their default values. In this means that a system reset automatically defines ROM bank 0 as a 32-Mbyte space with a start address of zero. This means that, except for ROM bank 0, all banks are undefined following a system startup.

The reset values for the next pointer and base pointer of ROM bank 0 are 0x200 and 0x000, respectively. This means that a system reset automatically defines ROM bank 0 as a 32-Mbyte space with a start address of zero. This initial definition of ROM bank 0 lets the system power-on or reset operation pass control to the user-supplied boot code that is stored in external ROM. (This code is located at address 0 in the system memory map.) When the boot code (i.e. ROM program) executes, it performs various system initialization tasks and reconfigures the system memory map according to the application's actual external memory and device configuration.

The initial system memory map following system startup is shown in following:

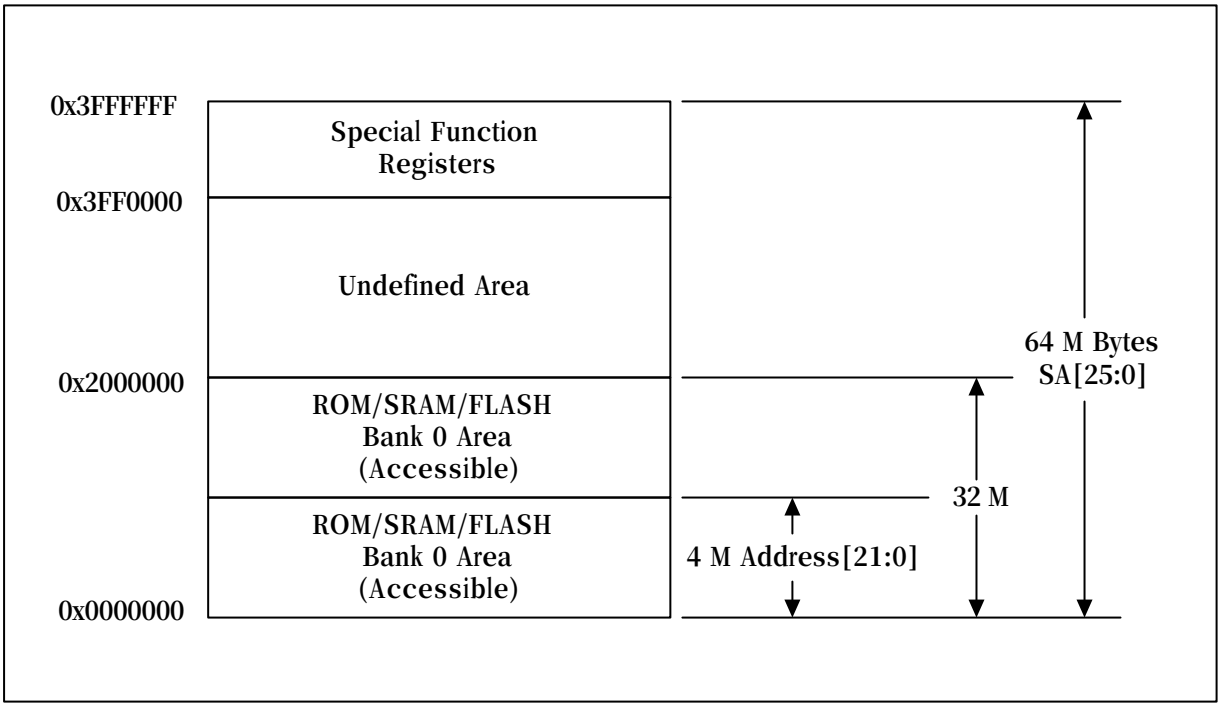

Figure 8 Initial system memory map (After reset)

6.2. Instruction / Data Cache

The S5N8947 CPU has a unified internal 8-Kbyte instruction/data cache. The cache is configured using two-way, set-associative addressing. The replacement algorithm is pseudo-LRU (Least Recently Used). The cache line size is four words (16 bytes). When a miss occurs, four words must be fetched consecutively from external memory. Typically, RISC processors take advantage of unified instruction/data caches to improve performance.

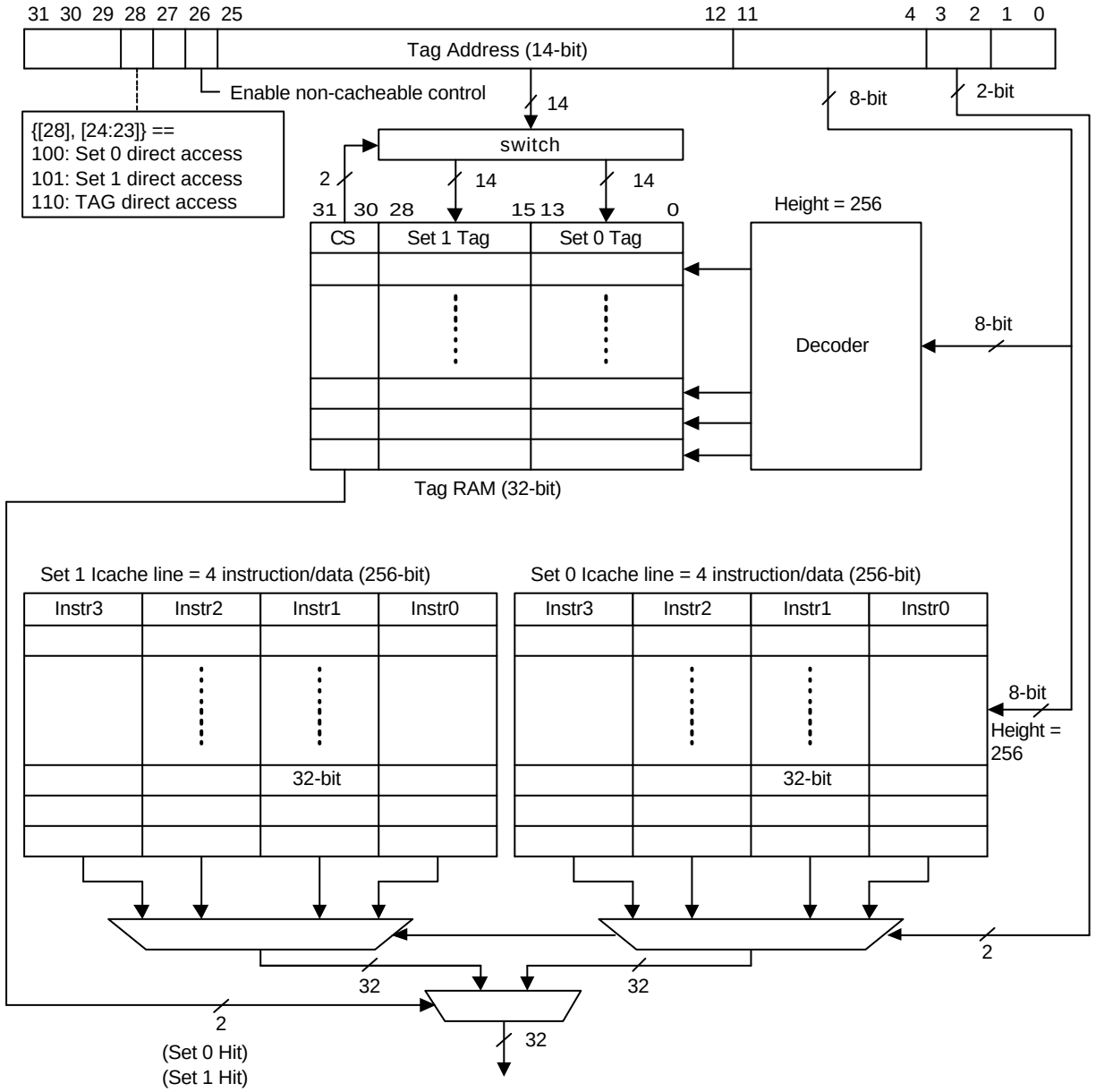

Figure 9 Memory Configuration for 8-Kbyte Cache

6.3. I²C Bus Controller

The S5N8947's Internal IC bus (I²C-bus) controller has the following important features:

- It requires only two bus lines, a serial data line (SDA) and a serial clock line (SCL). When the I²C-bus is free, both lines are High level.
- Each device that is connected to the bus is software-addressable by a unique address. Slave relationships on the bus are constant. The bus master can be either a master-transmitter or a master-receiver. The I²C bus controller supports only single master mode.
- It supports 8-bit, bi-directional, serial data transfers.
- The number of ICs that you can connect to the same I²C-bus is limited only by the maximum bus capacitance of 400 pF.

Following figure shows a block diagram of the S5N8947's I²C-bus controller.

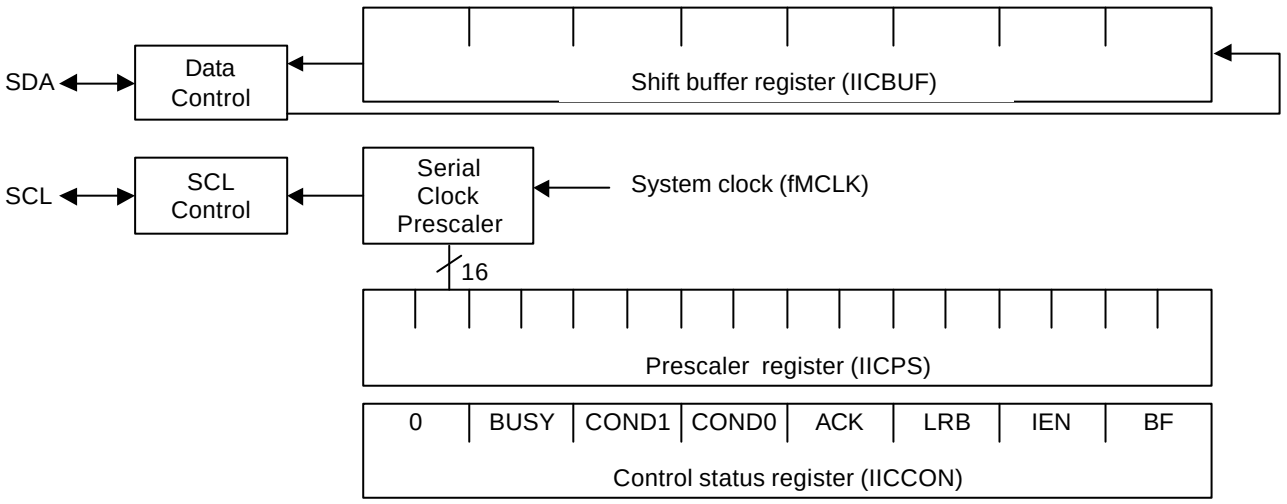

Figure 10 I²C-Bus block diagram

6.4. ETHERNET Controller

The S5N8947 has 2-channel Ethernet controller which operates at either 100/10-Mbits per second in half-duplex or full-duplex mode. In half-duplex mode, the controller supports the IEEE 802.3 Carrier Sense Multiple Access with Collision Detection (CSMA/CD) protocol. In full-duplex mode, it supports the IEEE 802.3 MAC Control Layer, including the Pause operation for flow control.

6.4.1. Block Diagram

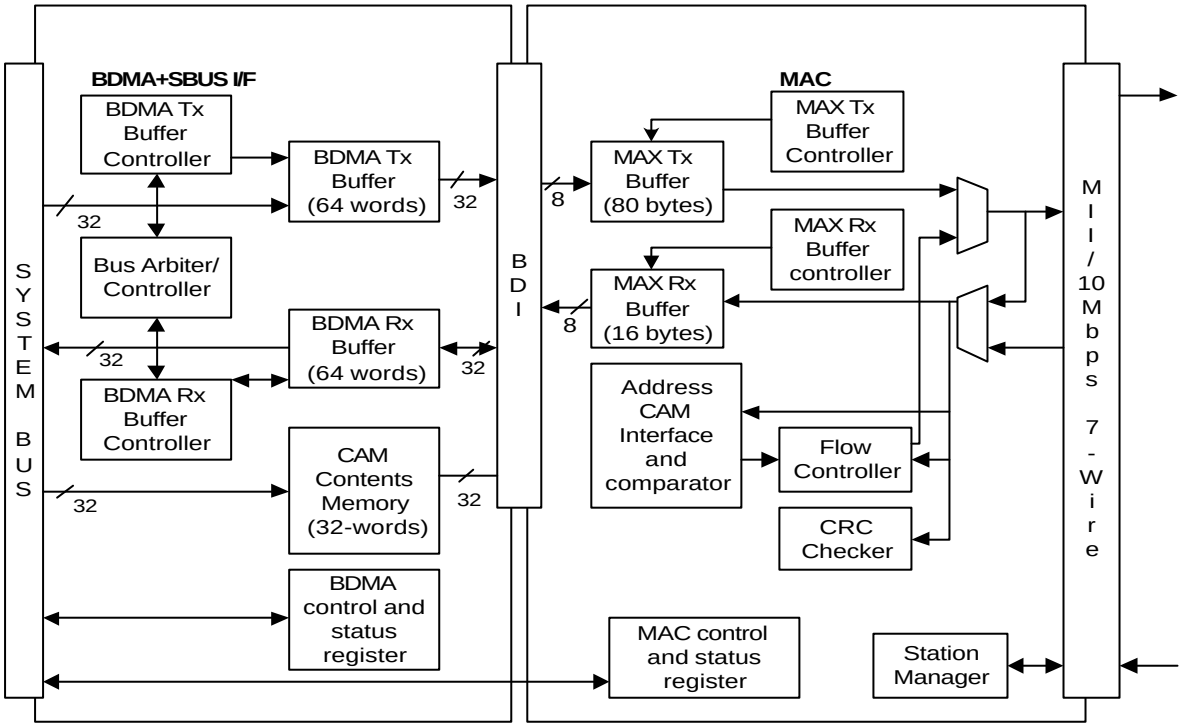

Figure 11 Ethernet controller block diagram

6.4.2. Features and Benefits

The most important features and benefits of the S5N8947 Ethernet controller are as follows:

- Cost-effective connection to an external Repeater Interface Controller(RIC)/Ethernet backbone
- Buffered DMA (BDMA) engine using Burst mode
- BDMA Tx/Rx buffers (256 bytes/256 bytes)
- MAC Tx/Rx FIFOs (80 bytes/16 bytes) to support re-transmit after collision without DMA request and to handle DMA latency
- Data alignment logic

- Supports for old and new media (compatible with existing 10-Mbit/s networks)
- Full IEEE 802.3 compatibility for existing applications
- Provides a standard Media Independent Interface (MII)
- Provides an external 7-wire interface, also.
- Station Management (STA) signaling for external physical layer configuration and link negotiation
- On-chip CAM (21 addresses)
- Full-duplex mode for doubled bandwidth
- Pause operation hardware support for full-duplex flow control
- Long packet mode for specialized environments
- Short packet mode for fast testing
- PAD generation for ease of processing and reduced processing time
- Support for old and new media : Compatible with existing 100/10Mbit/s networks.
- Full IEEE 802.3 compatibility : Compatible with existing hardware and software.
- Standard CSMA/CD,Full duplex capability at 100/10 Mbit/s : Increase in data throughput performance.

6.5. SAR and UTOPIA Interface

The S5N8947 provides ATM layer Segmentation and Reassembly (SAR) function over a 8bit UTOPIA interface. The S5N8947 delivers an integrated solution for performing the SAR tasks required to communicate over an ATM network. The device translates packet-based data into 53-byte ATM cells that are asynchronously mapped into various physical media. The S5N8947 can be effectively applied for equipment requiring an interface between packet-based data and ATM-based networks.

6.5.1. Block Diagram

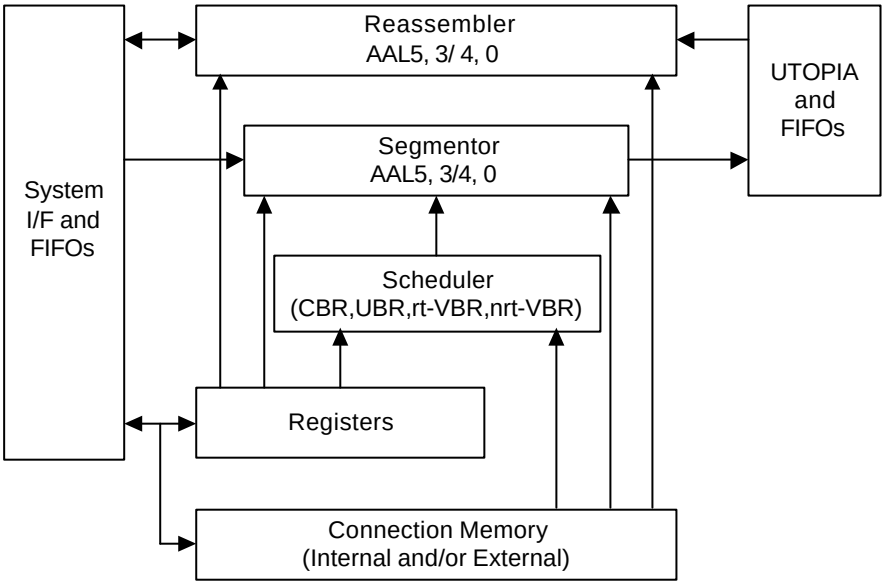

Figure 12 SAR function block diagram

6.5.2. Features and Benefits

- Supports CBR, UBR, rt-VBR and nrt-VBR traffic with rates set on a per-VC or per-VP basis.
- Supports AAL0 (raw cells) and AAL5 segmentation and reassembly.
- Segments and reassembles data up to about 70M bps via UTOPIA interface.
- Generates and verifies CRC-10 for OAM cells and AAL3/4 cells.
- Supports concurrent OAM cells and AAL5 cells on each active connection.
- Supports simultaneous segmentation and reassembly of up to 32 connections with internal memory and up to 4K connections with external memory.
- On chip 8K bytes SRAM for internal connection memory.
- Supports Contents Addressable Memory (CAM) for channel mapping (up to 32 connections).
- Supports packet sizes up to 64K bytes.
- Supports scatter and gather packet capability for large packets
- Start of Packet offset available for ease of implementing bridging and routing between different protocols.
- Provides glue-less UTOPIA level 2 interface (up to 3 PHYs).
- Supports big and little endian.

6.6. USB Controller

The Universal Serial Bus (USB) is an industry standard bus architecture for computer peripheral attachment. The USB provides a single interface for easy, plug-and-play, hot-plug attachment of peripherals such as keyboard, mouse, speakers, printers, scanners, and communication devices. The USB allows simultaneous use of many different peripherals with a combined transfer rate of up to 12 Mbit/s.

The S5N8947 controller includes a highly flexible integrated USB peripheral controller that lets designers implement a variety of microcontroller-based USB peripheral devices for telephony, audio, or other high-end applications. The S5N8947 controller is intended for USB peripherals that use the full-speed signalling rate of 12 Mbit/s. The USB low-speed rate (1.5 Mbit/s) is not supported. An integrated USB transceiver is provided to minimize system device count and cost. The USB peripheral controller's features meet or exceed all of the USB device class resource requirements defined by the USB specification Version 1.0 and 1.1. Consult the USB specification for details about overall USB system design. The integrated USB peripheral controller provides a very efficient and easy-to-use interface, so that device software (or firmware) does not incur the overhead of managing low-level USB protocol requirements.

The USB peripheral controller hardware implements a number of USB standard commands directly; the rest can be implemented in device software. In addition, the USB peripheral controller provides a high degree of flexibility to help designers accommodate vendor- or device-class-specific commands, as well as any new features that might be added in future USB specifications.

Specialized hardware is provided to support Bulk data transfers. Using the Microcontroller's DMA features, large size of bulk transfers from an off-chip peripheral, can be automatically synchronized to the USB data rate with little or no CPU overhead.

Robust error detection and management features are provided so the device software can manage transfers in any number of ways as required by the application. The USB suspend/resume, reset, and remote wake up features are also supported.

6.6.1. Block Diagram

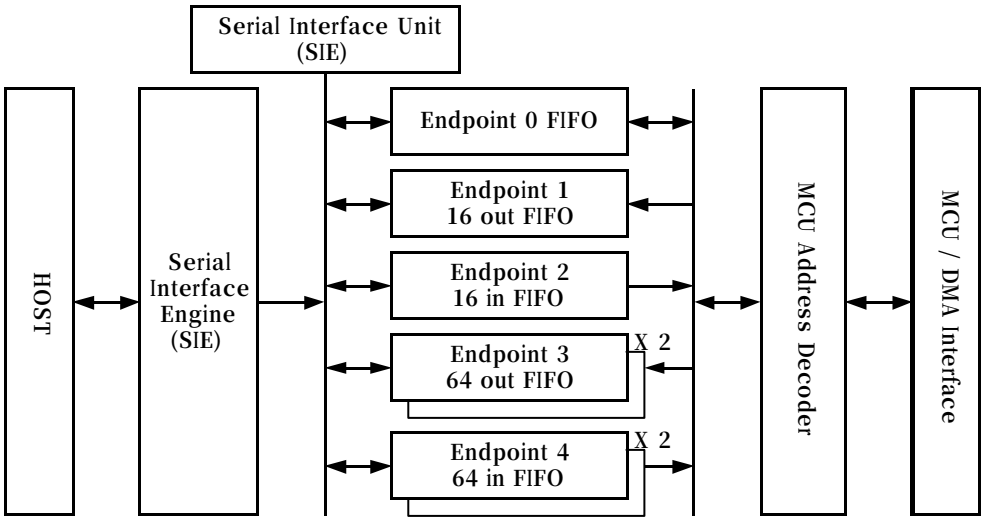

Figure 13 USB Module Block Diagram

6.7. DMA Controller

The S5N8947 has a two-channel general DMA controller, called the GDMA. The two-channel GDMA performs the following data transfers without CPU intervention:

- Memory-to-memory (memory to/from memory)
- UART-to-memory (serial port to/from memory)
- SPI-to-memory (SPI port to/from memory)

The on-chip GDMA can be started by software and/or by an external DMA request (nXDREQ). Software can also be used to restart a GDMA operation after it has been stopped.

The CPU can recognize when a GDMA operation has been completed by software polling and/or when it receives an appropriate internally generated GDMA interrupt. The S5N8947 GDMA controller can increment or decrement source or destination addresses and conduct 8-bit (byte), 16-bit (half-word), or 32-bit (word) data transfers.

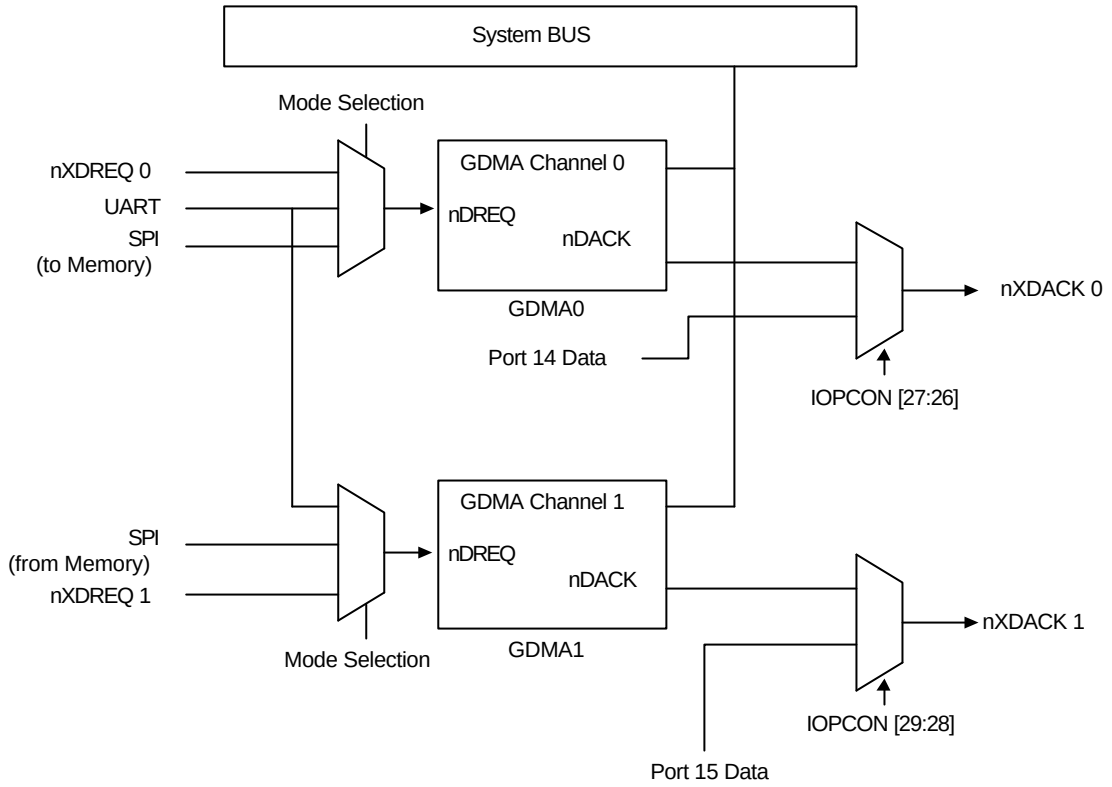

Figure 14 GDMA controller block diagram

6.8. UART

The S5N8947 Universal Asynchronous Receiver/Transmitter (UART) unit provides an asynchronous serial I/O (SIO) port. This can operate in interrupt-based or DMA-based mode. That is, the UART can generate internal interrupts or DMA requests to transfer data between the CPU and the serial I/O port.

The most important features of the S5N8947 UART include:

- Programmable baud rates
- Infra-red (IR) transmit/receive
- Insertion of one or two Stop bits per frame
- Selectable 5-bit, 6-bit, 7-bit, or 8-bit data transfers
- Parity checking

This unit has a baud rate generator, transmitter, receiver, and a control unit, as shown in next figure. The baud-rate generator can be driven by the internal system clock, MCLK. The transmitter and receiver block use this baud rate clock and have independent data buffer registers and data shifters.

Transmit data is written first to the transmit buffer register. From there, it is copied to the transmit shifter and then shifted out by the transmit data pin, UATXDn. Receive data is shifted in by the receive data pin, UARXDn. It is then copied from the shifter to the receive buffer register when one data byte has been received.

This unit provides software controls for mode selection, and for status and interrupt generation.

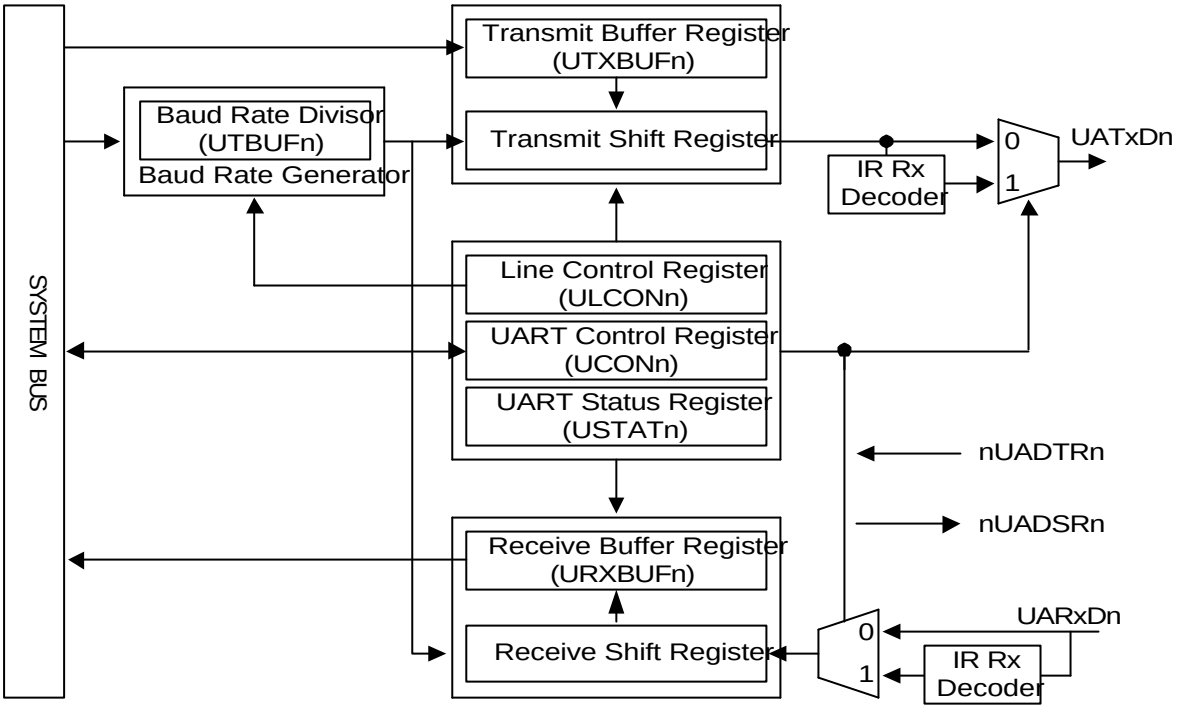

Figure 15 UART block diagram

6.9. Timers

The S5N8947 has three 32-bit timers. These timers can operate in interval mode or in toggle mode. The output signals are TOUT0 and TOUT1, respectively.

You enable or disable the timers by setting control bits in the timer mode register, TMOD. An interrupt request is generated whenever a timer count-out (down count) occurs.

Watchdog timer is also implemented in the S5N8947. The following guidelines apply to watchdog timer functions:

— When a watchdog timer is enabled, it loads a data value to its count register and begins decrementing the count register value by the system clock.

— If the reset from the watchdog timer (WDRESET) reaches to zero, the Watchdog will start its reset sequence. The reset value is then reloaded and the watchdog timer is disabled.

— The WDRESET performs the same function as the External Reset (System Reset) to each block.

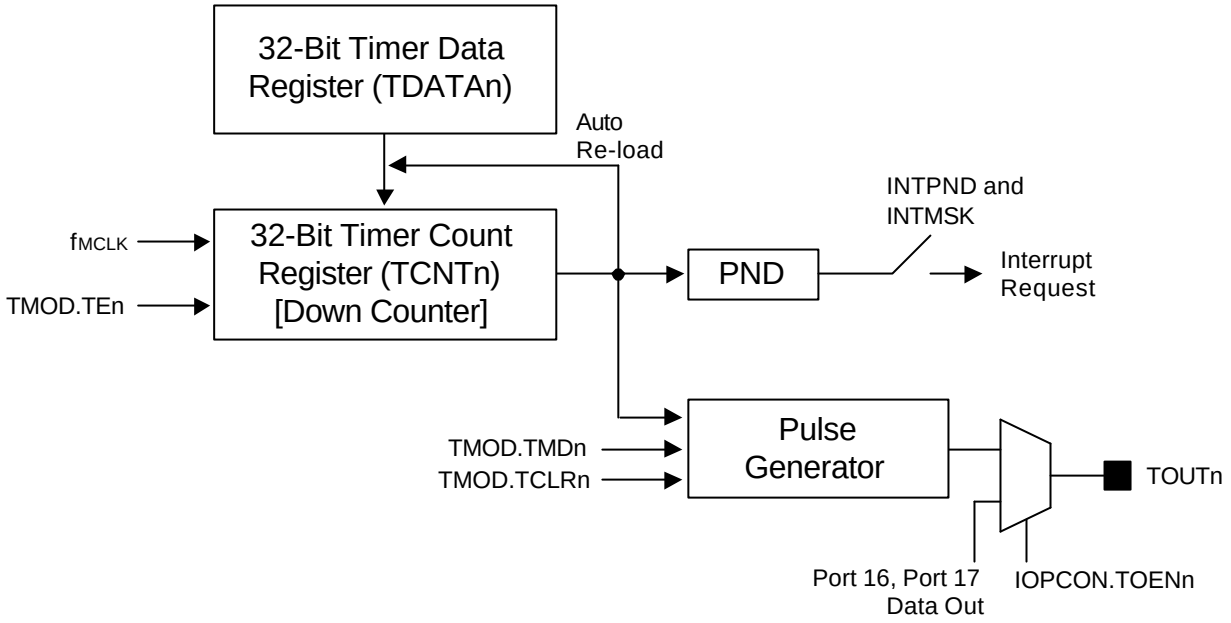

Figure 16 32-bit timer block diagram

6.10. I/O Ports

The S5N8947 has 18 programmable I/O ports. You can configure each I/O port to input mode, output mode, or special function mode. To do this, you write the appropriate settings to the IOPMOD, IOPCON0 and IOPCON1 registers. User can set filtering for the input ports using IOPCON0/1 register.

Port[0] can be used as nCE1 for PCMCIA interface or SPICLK, port[1] as nCE2 for PCMCIA interface or SPIMOSI, port[2] as nIOIS16 for PCMCIA interface or SPIMISO, port[3] as nALE for PCMCIA interface, port[4] as RW(external data transceiver direction) for PCMCIA interface, or port[7:5] as xINTREQ[2:0] depending on the settings in IOPCON0 register. And port[11:8] can be used as xINTREQ[6:3], port[13:12] as nXDREQ[1:0], port[15:14] as nXDACK[1:0], port[16] as TOUT0, or port[17] as TOUT1 depending on the settings in IOPCON1 register.

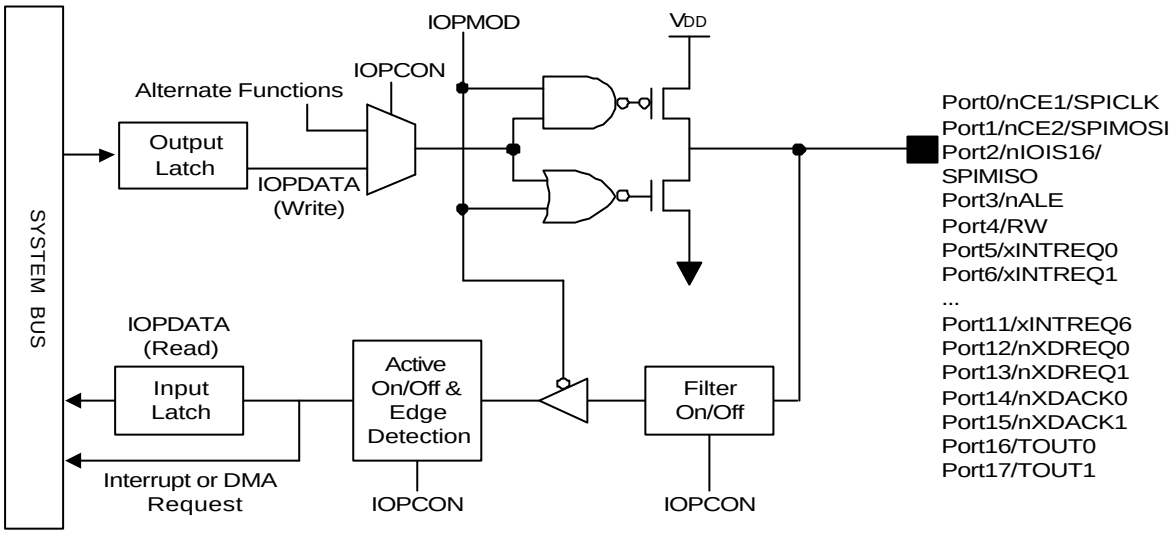

Figure 17 I/O port function diagram

6.11. Interrupt Controller

The S5N8947 interrupt controller has a total of 23 interrupt sources. Interrupt requests can be generated by internal function blocks and external pins.

The ARM7TDMI core recongnizes two kinds of interrupts: a normal interrupt request (IRQ), and a fast interrupt request (FIQ). Therefore all S5N8947 interrupts can be categorized as either IRQ or FIQ. The S5N8947 interrupt controller has an interrupt pending bit for each interrupt source.

Four special registers are used to control interrupt generation and handling:

- Interrupt priority registers. The index number of each interrupt source is written to the pre-defined interrupt priority register field to obtain that priority. The interrupt priorities are pre-defined from 0 to 22.
- Interrupt mode register. Defines the interrupt mode, IRQ or FIQ, for each interrupt source.
- Interrupt pending register. Indicates that an interrupt request is pending. If the pending bit is set, the interrupt pending status is maintained until the CPU clears it by writing a "1" to the appropriate pending register. When the pending bit is set, the interrupt service routine starts whenever the interrupt mask register is "0". The service routine must clear the pending condition by writing a "1" to the appropriate pending bit. This avoids the possibility of continuous interrupt requests from the same interrupt pending bit.

- Interrupt mask register. Indicates that the current interrupt has been disabled if the corresponding mask bit is "1". If an interrupt mask bit is "0" the interrupt will be serviced normally. If the global mask bit (bit 23) is set to "1", no interrupts are serviced. However, the source's pending bit is set if the interrupt is generated. When the global mask bit has been set to "0", the interrupt is serviced.

Index Values	Interrupt Sources
[22]	SPI interrupt
[21]	I^2C-bus interrupt
[20]	Ethernet controller 1 Rx interrupt
[19]	Ethernet controller 1 Tx interrupt
[18]	Ethernet controller 0 Rx interrupt
[17]	Ethernet controller 0 Tx interrupt
[16]	SAR Tx/Rx done interrupt
[15]	SAR Tx/Rx error interrupt
[14]	USB interrupt
[13]	GDMA channel 1 interrupt
[12]	GDMA channel 0 interrupt
[11]	Timer 2 interrupt
[10]	Timer 1 interrupt
[9]	Timer 0 interrupt
[8]	UART receive and error interrupt
[7]	UART transmit interrupt
[6]	External interrupt 6
[5]	External interrupt 5
[4]	External interrupt 4
[3]	External interrupt 3
[2]	External interrupt 2
[1]	External interrupt 1
[0]	External interrupt 0

Table 3 S5N8947 Interrupt Sources

6.12. SPI

The S5N8947 provides a Serial Peripheral Interface (SPI), which is used for register access of other devices, EEPROM and A/D converter. The S5N8947 SPI is full duplex, synchrounous channel and it consists of four signal, receive serial data, transmit serial data, clock and select. Inner baud rate generator create SPI clock and SPI signals are synchronized with this clock.

SPI can be operated with the help of GDMA. So multiple characters can be transmitted and received without host intervention. Otherwise, the host should transmit and receive individual character back-to-back with polling method.

SPI does not operate in slave mode and it also cannot be used for multimaster environment. It works with data characters from 4 to 32 bits long. Clock phase and polarity can be configured.

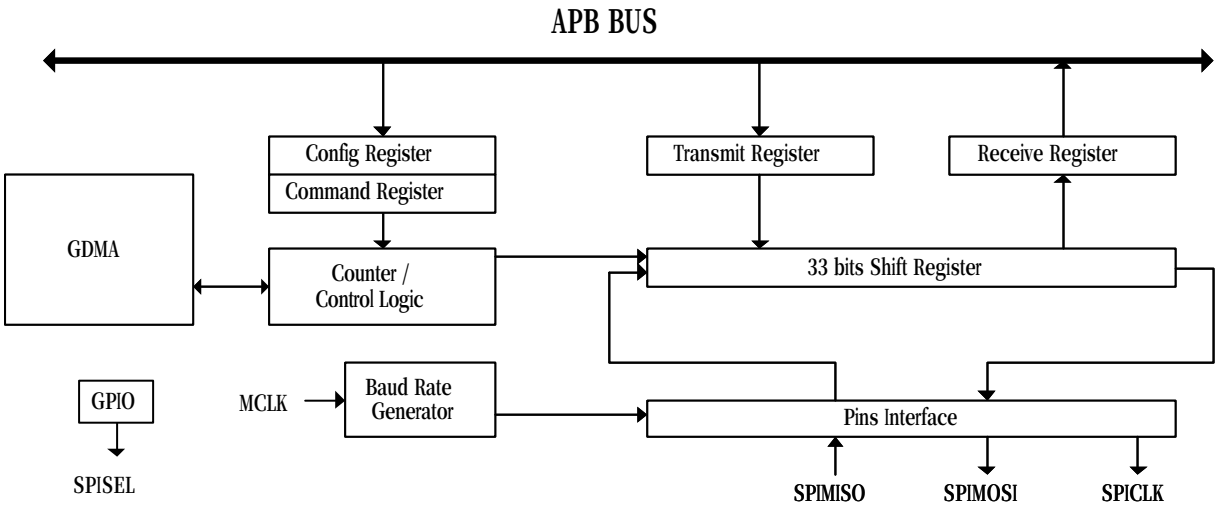

Figure 18 I/O Block diagram of SPI (Serial Peripheral Interface)

7. SPECIAL FUNCTION REGISTERS

Group	Registers	Offset	R/W	Description	Reset/Value
System Manager	SYSCFG	0x0000	R/W	System configuration register	0x23FF0000
	PCMCON	0x3000	R/W	PCMCIA Interface control register	0x80000000
	EXTACON0	0x3008	R/W	External I/O timing register 1	0x00000000
	EXTACON1	0x300C	R/W	External I/O timing register 2	0x00000000
	EXTDBWTH	0x3010	R/W	Data bus width for each memory bank	0x00000000
	ROMCON0	0x3014	R/W	ROM/SRAM/Flash bank 0 control register	0x20000060
	ROMCON1	0x3018	R/W	ROM/SRAM/Flash bank 1 control register	0x00000060
	ROMCON2	0x301C	R/W	ROM/SRAM/Flash bank 2 control register	0x00000060
	PCMOFFSET	0x3020	R/W	PCMCIA bank offset register	0x0000000
	DRAMCON0	0x3024	R/W	DRAM bank 0 control register	0x00000000
	DRAMCON1	0x3028	R/W	DRAM bank 1 control register	0x00000000
	DRAMCON2	0x302C	R/W	DRAM bank 2 control register	0x00000000
	DRAMCON3	0x3030	R/W	DRAM bank 3 control register	0x00000000
	REFEXTCON	0x3034	R/W	Refresh and external I/O control register	0x83FD0000
Ethernet1 (BDMA)	BDMATXCON	0x9000	R/W	Buffered DMA receive control register	0x00000000
	BDMARXCON	0x9004	R/W	Buffered DMA transmit control register	0x00000000
	BDMATXPTR	0x9008	R/W	Transmit trame descriptor start address	0x00000000
	BDMARXPTR	0x900C	R/W	Receive frame descriptor start address	0x00000000
	BDMARXLSZ	0x9010	R/W	Receive frame maximum size	Undefined
	BDMASTAT	0x9014	R/W	Buffered DMA status	0x00000000
	CAM	0x9100-0x917C	R/W	CAM content (32 words)	Undefined
	BDMATXBUF	0x9200-0x92FC	R/W	BDMA Tx buffer (64 words) for test mode addressing	Undefined
	BDMARXBUF	0x9800-0x99FC	R/W	BDMA Rx buffer (64 words) for test mode addressing	Undefined
Ethernet1 (MAC)	MACON	0xA000	R/W	Ethernet MAC control register	0x00000000
	CAMCON	0xA004	R/W	CAM control register	0x00000000
	MACTXCON	0xA008	R/W	MAC transmit control register	0x00000000
	MACTXSTAT	0xA00C	R/W	MAC transmit status register	0x00000000
	MACRXCON	0xA010	R/W	MAC receive control register	0x00000000
	MACRXSTAT	0xA014	R/W	MAC receive status register	0x00000000
	STADATA	0xA018	R/W	Station management data	0x00000000
	STACON	0xA01C	R/W	Station management control and address	0x00006000
	CAMEN	0xA028	R/W	CAM enable register	0x00000000
	EMISSCNT	0xA03C	R/W	Missed error count register	0x00000000
	EPZCNT	0xA040	R	Pause count register	0x00000000
	ERMPZCNT	0xA044	R	Remote pause count register	0x00000000
	ETXSTAT	0x9040	R	Transmit control frame status	0x00000000
Ethernet2 (BDMA)	BDMATXCON	0xE000	R/W	Buffered DMA receive control register	0x00000000
	BDMARXCON	0xE004	R/W	Buffered DMA transmit control register	0x00000000
	BDMATXPTR	0xE008	R/W	Transmit trame descriptor start address	0x00000000
	BDMARXPTR	0xE00C	R/W	Receive frame descriptor start address	0x00000000

	BDMARXLSZ	0xE010	R/W	Receive frame maximum size	Undefined
	BDMASTAT	0xE014	R/W	Buffered DMA status	0x00000000
	CAM	0xE100-0xE17C	R/W	CAM content (32 words)	Undefined
	BDMATXBUF	0xE200-0xE2FC	R/W	BDMA Tx buffer (64 words) for test mode addressing	Undefined
	BDMARXBUF	0xE800-0xE9FC	R/W	BDMA Rx buffer (64 words) for test mode addressing	Undefined
Ethernet2 (MAC)	MACON	0xF800	R/W	Ethernet MAC control register	0x00000000
	CAMCON	0xF804	R/W	CAM control register	0x00000000
	MACTXCON	0xF808	R/W	MAC transmit control register	0x00000000
	MACTXSTAT	0xF80C	R/W	MAC transmit status register	0x00000000
	MACRXCON	0xF810	R/W	MAC receive control register	0x00000000
	MACRXSTAT	0xF814	R/W	MAC receive status register	0x00000000
	STADATA	0xF818	R/W	Station management data	0x00000000
	STACON	0xF81C	R/W	Station management control and address	0x00006000
	CAMEN	0xF828	R/W	CAM enable register	0x00000000
	EMISSCNT	0xF83C	R/W	Missed error count register	0x00000000
	EPZCNT	0xF840	R	Pause count register	0x00000000
	ERMPZCNT	0xF844	R	Remote pause count register	0x00000000
	ETXSTAT	0xE040	R	Transmit control frame status	0x00000000
USB	FA	0x7000	R/W	Function address register	0x00000000
	PM	0x7004	R/W	Power/System management register	0x00000000
	INT	0x7008	R/W	Interrupt register	0x00000000
	INTE	0x700C	R/W	Interrupt Enable register	0x0000041F
	FN	0x7010	R	Frame Number register	0x00000000
	E0SC	0x7014	R/W	Endpoint 0 Status Control register	0x00005080
	E0SA	0x7018	R/W	Endpoint 0 DMA Start Address register	0x00000000
	E0XDS	0x701C	R/W	Endpoint 0 Receive/Transmit Data Size register	0x00000000
	E0LDS	0x7020	R/W	Endpoint 0 Limit Data Size register	0x00800000
	E1SC	0x7024	R/W	Endpoint 1 Status Control register	0x00000100
	E1SA	0x7028	R/W	Endpoint 1 DMA Start Address register	0x00000000
	E1RDS	0x702C	R/W	Endpoint 1 Transmit Data Size register	0x00000000
	E1LDS	0x7030	R/W	Endpoint 1 Limit Data Size register	0x00800000
	E2SC	0x7034	R/W	Endpoint 2 Status Control register	0x00005080
	E2SA	0x7038	R/W	Endpoint 2 DMA Start Address register	0x00000000
	E2TDS	0x703C	R/W	Endpoint 2 Transmit Data Size register	0x00000000
	E3SC	0x7040	R/W	Endpoint 3 Status Control register	0x00000004
	E3SA	0x7044	R/W	Endpoint 3 DMA Start Address register	0x00000000
	E3RDS	0x7048	R/W	Endpoint 3 Transmit Data Size register	0x00000000
	E3LDS	0x704C	R/W	Endpoint 3 Limit Data Size register	0x00800000
	E4SC	0x7050	R/W	Endpoint 4 Status Control register	0x00005080
	E4SA	0x7054	R/W	Endpoint 4 DMA Start Address register	0x00000000
	E4TDS	0x7058	R/W	Endpoint 4 Transmit Data Size register	0x00000000
SAR	SW_RESET	0x8000	R/W	Software reset register	0x00000000
	GLOBAL_MODE	0x8008	R/W	Global mode register	0x00000000
	TIMEOUT_BASE	0x800C	R/W	Base multiple for receive packet timeout register	0x00FF7FFF
	TX_READY1	0x8010	R/W	Transmit ready first packet or subpacket address	0x00000000
	TX_READY2	0x8014	R/W	Transmit ready last packet or subpacket address	0x00000000
	TX_DONE_ADDR	0x8018	R/W	Transmit packet done queue base address register	0x00000000
	TX_DONE_SIZE	0x801C	R/W	Transmit packet done queue size register	0x00C00000
	RX_POOL0_ADDR	0x8020	R/W	Receive queue 0 base address register	0x00000000
	RX_POOL0_SIZE	0x8024	R/W	Receive queue 0 size register	0x00C00000
	RX_POOL1_ADDR	0x8028	R/W	Receive queue 1 base address register	0x00000000

	RX_POOL1_SIZE	0x802C	R/W	Receive queue 1 size register	0x00C00000
	RX_POOL2_ADDR	0x8030	R/W	Receive queue 2 base address register	0x00000000
	RX_POOL2_SIZE	0x8034	R/W	Receive queue 2 size register	0x00C00000
	RX_POOL3_ADDR	0x8038	R/W	Receive queue 3 base address register	0x00000000
	RX_POOL3_SIZE	0x803C	R/W	Receive queue 3 size register	0x00C00000
	RX_DONE0_ADDR	0x8040	R/W	Receive packet done queue 0 base address register	0x00000000
	RX_DONE0_SIZE	0x8044	R/W	Receive packet done queue 0 size register	0x00C00000
	RX_DONE1_ADDR	0x8048	R/W	Receive packet done queue 1 base address register	0x00000000
	RX_DONE1_SIZE	0x804C	R/W	Receive packet done queue 1 size register	0x00C00000
	UTOPIA_CONFIG	0x8050	R/W	UTOPIA interface configuration register	0x00C00000
	UTOPIA_TIMEOUT	0x8054	R/W	UTOPIA interface timeout register	0xFFFFFFFF
	CLOCK_RATIO	0x8064	R/W	Ratio of SAR clock freq toUNI interface speed	0x0000008E
	DONE_INT_MASK	0x8070	R/W	Interrupt mask for done interrupt register	0xFFFFFFFF
	ERR_INT_MASK	0x8074	R/W	Interrupt mask for error interrupt register	0xFFFFFFFF
	DONE_INT_STAT	0x8078	R/W	Interrupt status for done interrupt register	0x00000000
	ERR_INT_STAT	0x807C	R/W	Interrupt status for error interrupt register	0x00000000
	1/R_LOOKUP_TBL	0x8080	R/W	Base address of 1/Rate lookup table	0x00000000
	VP_LOOKUP_TBL	0x8084	R/W	Base address of VP lookup table	0x00200000
	UBR_SCH_TBL	0x8088	R/W	Base address and entry number of UBR schedule	0x0030007F
	CBR_SCH_TBL	0x808C	R/W	Base address and entry number of CBR schedule	0x0038007F
	CELL_BUFF	0x8090	R/W	Base address and entry number of cell buffer	0x0040000F
	SCH_CONN_TBL	0x8094	R/W	Base address and entry number of scheduler connection table	0x0050001F
	AAL_CONN_TBL	0x8098	R/W	Base address and entry number of AAL connection table	0x0060001F
	SAR_CONN_TBL	0x809C	R/W	Base address and entry number of SAR connection table	0x00700000
	CAM_VPVC/CN	0x8100-0x81FC	R/W	CAM VPCI, VCI and connection number register	0x00000000
	CONFIGURATION	0x8200	R/W	Clock control and connection memory configuration register	0x00000046
	EXT_CMBASE	0x8204	R/W	External connection memory base address register	0x00000000
I/O Ports	IOPMOD	0x5000	R/W	I/O port mode register	0x00000000
	IOPCON0	0x5004	R/W	I/O port control 0 register	0x00000000
	IOPCON1	0x5008	R/W	I/O port control 1 register	0x00000000
	IOPDATA	0x500C	R/W	I/O port data register	Undefined
SPI	SPICFG	0x5804	R/W	SPI configuration register	0x0000000F
	SPISTS	0x5808	R	SPI status register	0x00000000
	SPICMD	0x580C	R/W	SPI command register	0x00000000
	TXCHR	0x5810	R/W	SPI transmit register	0x00000000
	RXCHR	0x581C	R	SPI receive register	0x00000000
Interrupt Controller	INTMOD	0x4000	R/W	Interrupt mode register	0x00000000
	INTPND	0x4004	R/W	Interrupt pending register	0x00000000
	INTMSK	0x4008	R/W	Interrupt mask register	0x00FFFFFF
	INTPRI0	0x400C	R/W	Interrupt priority register 0	0x03020100
	INTPRI1	0x4010	R/W	Interrupt priority register 1	0x07060504
	INTPRI2	0x4014	R/W	Interrupt priority register 2	0x0B0A0908
	INTPRI3	0x4018	R/W	Interrupt priority register 3	0x0F0E0D0C
	INTPRI4	0x401C	R/W	Interrupt priority register 4	0x13121110
	INTPRI5	0x4020	R/W	Interrupt priority register 5	0x00161514
	INTOFFSET	0x4024	R	Interrupt offset address register	0x0000005C
	INTPNDPRI	0x4028	R	Interrupt pending priority register	0x00000000
	INTPNDTST	0x402C	W	Interrupt pending test register	0x00000000

	INTOSET_FIQ	0x4030	R	FIQ interrupt offset register	0x0000005C
	INTOSET_IRQ	0x4034	R	IRQ interrupt offset register	0x0000005C
I ^{2}C Bus	IICCON	0XF000	R/W	I^{2}C bus control status register	0x00000000
	IICBUF	0xF004	R/W	I^{2}C bus shift buffer register	Undefined
	IICPS	0xF008	R/W	I^{2}C bus prescaler register	0x00000000
	IICCOUNT	0xF00C	R	I^{2}C bus prescaler counter register	0x00000000
GDMA	GDMACON0	0xB000	R/W	GDMA channel 0 control register	0x00000000
	GDMACON1	0xC000	R/W	GDMA channel 1 control register	0x00000000
	GDMASRC0	0xB004	R/W	GDMA source address register 0	Undefined
	GDMADST0	0xB008	R/W	GDMA destination address register 0	Undefined
	GDMASRC1	0xC004	R/W	GDMA source address register 1	Undefined
	GDMADST1	0xC008	R/W	GDMA destination address register 1	Undefined
	GDMACNT0	0xB00C	R/W	GDMA channel 0 transfer count register	Undefined
	GDMACNT1	0xC00C	R/W	GDMA channel 1 transfer count register	Undefined
UART	ULCON	0xD000	R/W	UART line control register	0xXXXXXX00
	UCON	0xD004	R/W	UART control register	0xXXXXXX00
	USTAT	0xD008	R	UART status register	0xXXXXXXC0
	UTXBUF	0xD00C	W	UART transmit holding register	Undefined
	URXBUF	0xD010	R	UART receive buffer register	Undefined
	UBRDIV	0xD014	R/W	Baud rate divisor register	0xXXXXXX00
Timers	TMOD	0x6000	R/W	Timer mode register	0x00000000
	TDATA0	0x6004	R/W	Timer 0 data register	0x00000000
	TDATA1	0x6008	R/W	Timer 1 data register	0x00000000
	TDATA2	0x600C	R/W	Timer 2 data register	0x00000000
	TCNT0	0x6010	R/W	Timer 0 count register	0xFFFFFFFF
	TCNT1	0x6014	R/W	Timer 1 count register	0xFFFFFFFF
	TCNT2	0x6018	R/W	Timer 2 count register	0xFFFFFFFF
	WDCON	0x601C	R/W	Watchdog Timer Control register	0xFFFFF00
	WDCNT	0x6020	R	Watchdog Timer Count register	0xFFFFFFFF

8. ELECTRIC CHARACTERISTICS

8.1. Absolute Maximum Ratings

Parameter	Symbol	Rating		Units
Supply Voltage	V_{DD}	1.8V V_{DD}	2.7	V
		3.3V V_{DD}	3.8	
DC input Voltage	V_{IN}	1.8V input buffer	2.7	V
		1.8V interface 3.3V tolerant input buffer	3.8	
Operating temperature	T_{OPR}	– 40 to 85		°C
Storage temperature	T_{STG}	– 65 to 150		°C

Table 4 Absolute Maximum Ratings

8.2. Recommended Operating Conditions

Parameter	Symbol	Rating		Units
Supply Voltage	V_{DD}/V_{DDA}	1.8V V_{DD}	1.8 ± 0.15	V
		3.3V V_{DD}	3.3 ± 0.3	
Oscillator frequency	f_{OSC}	12		MHz
External Loop Filter Capacitance	L_F	320		pF
Industrial Temperature Range	T_A	-40 to 85		°C

Table 5 Recommaended Operating Conditions

Parameter	Symbol	Min	Typ	Max	Units
Power Dissipation	P_D		300		mW

Table 6 Power Dissipation

NOTES

✓ It is strongly recommended that all the supply pins (V_{DD}/V_{DDA}) be powered from the same source to avoid power latch-up.

8.3. DC Electrical Characteristics

V_{DD} = 1.8 ± 0.15 V, V_{EXT} = 3.0 ± 0.3 V, T_A = -40 to 85 C (in case of 3.3 V-tolerant I/O)

Parameter		Symbol	Conditions	Min	Typ	Max	Unit
High level input voltage	LVCMOS i/f	V_{IH}	–	1.27	–	–	V
Low level input voltage	LVCMOS i/f	V_{IL}	–	–	–	0.57	V
Switching threshold		VT	LVCMOS	–	0.55V_{DD}	–	V
Schmitt trigger positive-going threshold		VT+	LVCMOS	–	–	1.27	–
Schmitt trigger negative-going threshold		VT–	LVCMOS	0.57	–	–	–
High level input current	Input buffer	I_{IH}	$V_{IN} = V_{DD}$	– 10	–	10	A
	Input buffer with pull-up			5	18	40	
Low level input current	Input buffer	I_{LH}	$V_{IN} = V_{SS}$	– 10	–	10	A
	Input buffer with pull-up			– 40	– 18	– 5	
High level output voltage	Type B1 to B12	V_{OH}	I_{OH} = – 1 A	V_{DD} – 0.05	–	–	V
	Type B1		I_{OH} = – 1 mA	1.2			
	Type B2		I_{OH} = – 2 mA				
	Type B4		I_{OH} = – 4 mA				
	Type B6		I_{OH} = – 6 mA				
Low level output voltage	Type B1 to B12	V_{OL}	I_{OL} = 1 A			0.05	V
	Type B1		I_{OL} = 1 mA			0.45	
	Type B2		I_{OL} = 2 mA				
	Type B4		I_{OL} = 4 mA				
	Type B6		I_{OL} = 6 mA				
Tri-state output leakage current		I_{OZ}	$V_{OUT} = V_{SS}$ or V_{DD}	– 10		10	A
Maximum operating current		I_{DD}	V_{DD} = 3.6 V, f_{MCLK} = 50MHz			100	A

Table 7 DC Electrical Characteristics

9. PACKAGE DIMENSION

This section describes the mechanical data for the S5N8947 208-pin LQFP package.

208-LQFP-2828 PACKAGE DIMENSIONS

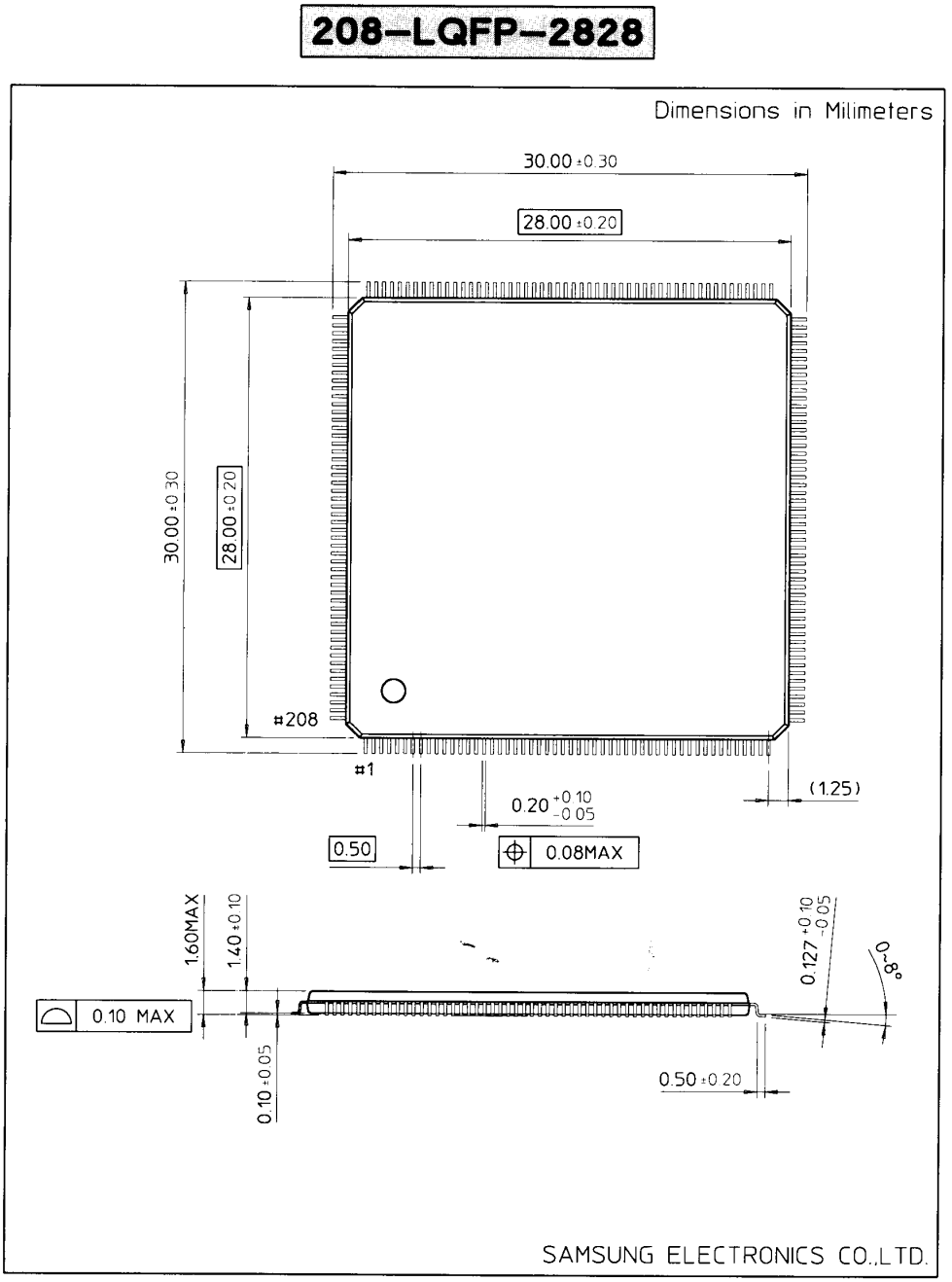

Figure 19 208-LQFP-2828 Package Dimensions