MT90500 Replacement Guidelines

Application Note

March 2003

Contents

Purpose

This document will describe the transition of a design from the MT90500 to the MT90503. The document will provide an overview of the replacement of an MT90500 with an MT90503, using the MT90503 to provide the same functionality as the MT90500.

Scope

The scope of this document will cover schematic level hardware transition and API level software transition. This document is not intended to provide a feature comparison or operational comparison of the two SARs. This document is not intended to show the additional functionality or the benefits of the MT90503 over the MT90500.

1.0 Introduction

The MT90503 AAL1 SAR is the recommended replacement in designs currently using the MT90500 AAL1 SAR. The following two sections, Hardware Considerations and Software Considerations, will provide guidelines for replacing an MT90500 with an MT90503. The intention of this document is to show how the MT90503 may be used to provide the same functionality as available on the MT90500. It is not the intention of this document to show the increased functionality and features of the MT90503 such as the increased channel capacity or support for UTOPIA Level 2 interface. Full feature lists, data sheets and a selector guide comparison are available on the Zarlink Semiconductor web site.

2.0 Hardware Considerations

The following sections provide pin to pin comparisons of the MT90500 and MT90503. Each individual section will show the current MT90500 pins used to provide the applicable interface and the equivalent pins (if any) on the MT90503.

2.1 Package

The MT90500 is a 240-pin PQFP package. The MT90503 is a 503-ball PBGA package. The footprint area of the MT90500 is 1197.16 mm^2. The footprint area of the MT90503 is 1600 mm^2. The mechanical drawings and dimensions are available in each chip's respective data sheet in the Physical Specification section.

2.2 Power and Ground

The MT90500 has a core power rail of 3.3V and I/O power rails of 3.3V and 5V. The MT90503 has core and I/O power rails of 3.3V. The 3.3V rail of the MT90500 typically uses 400 mA of current when using a 60 MHz master clock. The 3.3V rail of the MT90503 typically uses 720 mA of current for 1024 channels when using a 50 MHz master clock (see MT90503 data sheet DC Characteristics for full details).

The power sequencing for the MT90503 requires the chip to be held in reset until the power rails are stable or have its global_tri_state pin held asserted. The MT90500 recommended the TRISTATE pin be held asserted during power-up. These pins are listed in the Miscellaneous section.

Although the MT90503 does not require a 5V I/O power rail, the TDM interface does have the option of using a 5V I/O supply. This is described in Section 2.6 “TDM Interface” on page 19.

MT90500 Pin Name	MT90500 Pin Number	MT90503 Equivalent Pin Name	MT90503 Equivalent Pin Number	Action
IO_VSS	1, 7, 16, 29, 43, 61, 86, 91, 110, 119, 129, 139, 151, 163, 172, 182, 197, 213, 229	GND	A2, A4, A7, A8, A9, A12, A15, A18, A21, A22, A23, A25, A26, A28, B1, B29, D1, D26, D29, E1, E6, E29, G1, G29, H1, H29, J1, J29, L11, L13, L15, L17, L19, M1, M29, N11, N13, N15, N17, N19, R1, R11, R13, R17, R19, R29, T29, U11, U13, U15, U17, U19, V1, W11, W13, W15, W17, W19, AA1, AA29, AB1, AB29, AC1,AD5, AD25, AE1, AE13, AE29, AF1, AF29, AG29, AH1, AH2, AH28, AJ2, AJ4, AJ7, AJ8, AJ9, AJ12, AJ15, AJ18, AJ21, AJ22, AJ23, AJ25, AJ26, AJ28	None
CORE_VSS	100, 141, 161			
RING_VSS	20, 40, 80, 201, 221			
IO_VDD_3V	92, 111, 120, 132, 145, 157, 169, 181	VDD	A3, A5, A10, A11, A14, A16, A19, A20, A27, A29, B2, C1, C29, K1, K29, L1, L29, P1, P29, T1, V29, W1, W29, Y1, Y29, AC29, AG1, AH3, AH29, AJ1, AJ3, AJ5, AJ10, AJ11, AJ14, AJ16, AJ19, AJ20, AJ27, AJ29	None
CORE_VDD_3V	101, 140, 160			
RING_VDD_3V	21, 41, 81, 200, 220			
IO_VDD_5V	2, 13, 24, 42, 60, 88, 183, 207, 225, 240			None

Table 1 - Power and Ground Pins

2.3 CPU Interface

The MT90500 supports a Motorola or Intel interface, using 16-bit (word) data accesses. The address placed on the address bus is a byte address, hence the MT90500 has no A0 pin. The MT90500 uses what the MT90503 refers to as 16-bit data bus, direct, non-multiplexed mode of operation.

The MT90500 has a separate memory map for internal and external memory. The selection of internal or external memory access is indicated explicitly via the AEM pin. The MT90503 has a single memory map for both internal and external memory. The locations are distinguished based on the address within the memory map. The MT90503 memory map is listed in the Memory Map section of its data sheet.

The MT90500 has a 5V interface whereas the MT90503 has a 3.3V interface that is 5V tolerant.

MT90500 Pin Name	MT90500 Pin Number	MT90503 Equivalent Pin Name	MT90503 Equivalent Pin Number	Action
Intel/$\overline{\text{Motorola}}$	37	cpu_mode[2]	AH22	None
IC	36			None
$\overline{\text{CS}}$	203	$\overline{\text{inmo\_cs}}$	AH17	None
$\overline{\text{WR}}$/R \ $\overline{\text{W}}$	237	inmo_$\overline{\text{wr}}$_r/$\overline{\text{w}}$	AF17	None
$\overline{\text{RD}}$/$\overline{\text{DS}}$	239	$\overline{\text{inmo\_rd\_ds}}$	AH16	None
RDY/$\overline{\text{DTACK}}$	238	inmo_rdy/$\overline{\text{ndtack}}$	AG16	None
$\overline{\text{INT}}$	84	interrupt1	AH11	None
D[15:0]	223, 222, 219, 218, 217, 216, 215, 214, 212, 211, 210, 209, 208, 206, 205, 204	inmo_d[15:8]	AE12, AF12, AG12, AH12, AF13, AG13, AH13, AJ13	None
		inmo_d[7:0]	AE14, AF14, AG14, AH14, AG15, AH15, AE16, AF16	None
AEM	184			None
A[15:1]	185, 186, 197, 188, 189, 190, 191, 192, 193, 194, 195, 196, 198, 199, 202	inmo_a[14:0]	AE18, AF18, AG18, AH18, AE19, AF19, AG19, AH19, AF20, AG20, AH20, AE21, AF21, AG21, AH21	None
		inmo_a_das	AJ17	Usually connected to upper CPU address line. See MT90503 data sheet CPU Interface section for details.
		inmo_ale	AG17	Tie high (unused)
		cpu_mode[0]	AE20	Tie low (non-multiplexed)
		cpu_mode[1]	AG22	Tie low (16-bit data bus)
		cpu_mode[3]	AF22	Tie low
		interrupt2	AG11	No connect

Table 2 - CPU Interface Pins

2.4 UTOPIA Interface

The MT90500 has one and a half UTOPIA ports, known as the Primary and Secondary ports. The MT90503 has three UTOPIA ports, known as Port A, Port B and Port C. The MT90503 has the capability to switch cells between each of its three UTOPIA ports. That is to say that a cell received on Port A may be transmitted out Port A, Port B and/or Port C. The same functionality exists for cells received on Port B or Port C.

The MT90503 is able to duplicate the operation of the MT90500 covered by several operational configurations for the current MT90500 application. These configurations are based on Figure 63 on page 139 of the MT90500 data sheet, which is shown with less detail below.

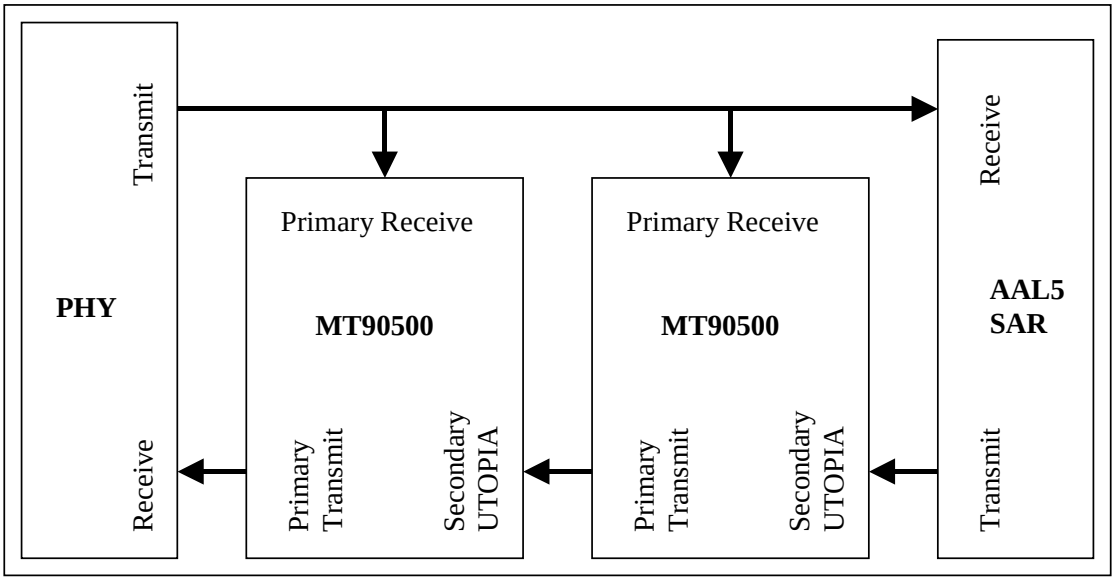

Figure 1 - MT90500 UTOPIA Interconnections

The MT90500 has a 5V interface whereas the MT90503 has a 3.3V interface that is 5V tolerant. The MT90503 specifies that its output clav signal must be pulled down externally, while its output enb signal must be pulled up externally.

2.4.1 PHY & MT90500

In this configuration the MT90500 is interfaced to a PHY device via its Primary port. There are no other devices present on the bus (such as a second MT90500 or an AAL5 SAR). In this setup it is likely that the PRXEN of the MT90500 and the enb of the PHY are both grounded. The MT90500 is operating in 8-bit ATM mode. The MT90503 will use Port A, configured for 8-bit, Level 1, ATM mode.

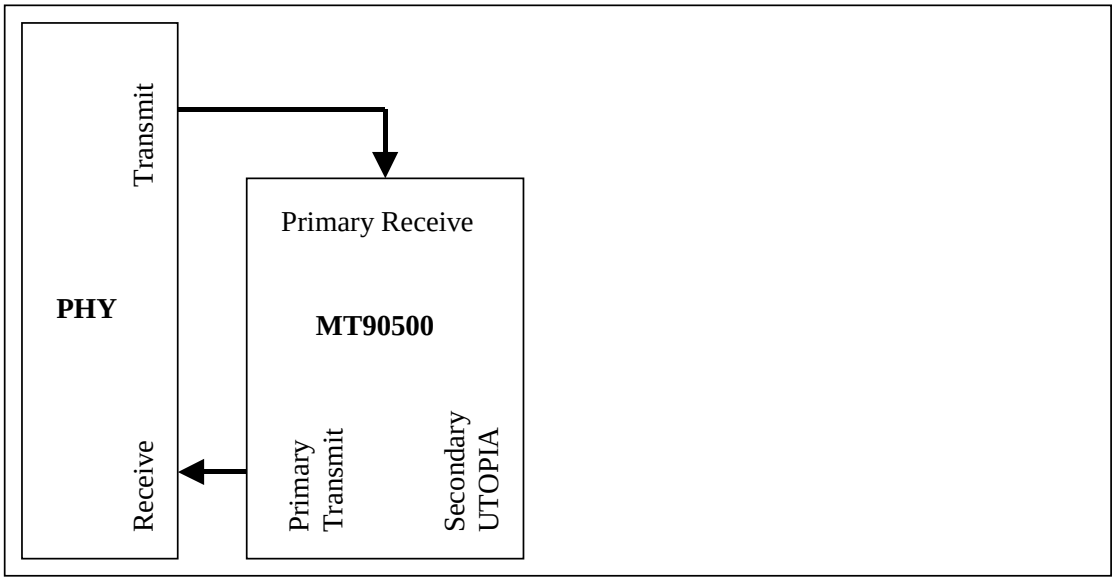

Figure 2 - PHY & MT90500

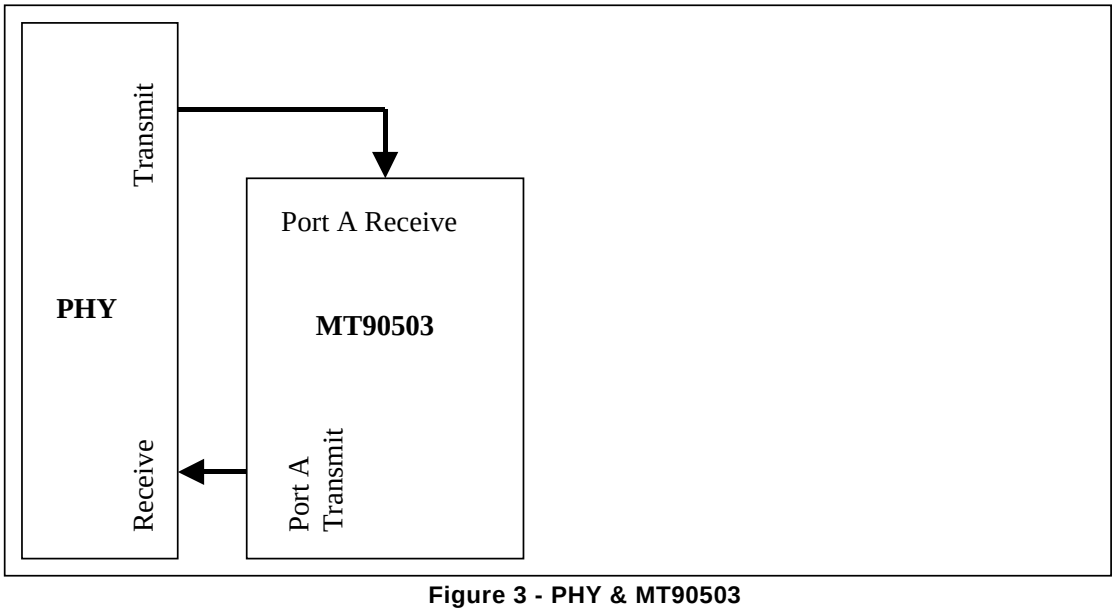

Figure 3 - PHY & MT90503

In this configuration Port B and Port C of the MT90503 are unused. See sections 2.4.4 MT90503 Port B Unused and 2.4.5 MT90503 Port C Unused for details.

MT90500 Pin Name	MT90500 Pin Number	MT90503 Equivalent Pin Name	MT90503 Equivalent Pin Number	Action
PTXDATA[7:0]	49, 48, 47, 46, 45, 44, 39, 38	txa_data[7:0]	L5, K2, K3, K4, K5, J2, J3, J4	None
PTXSOC	52	txa_soc	N2	None
$\overline{\text{PTXEN}}$	51	$\overline{\text{txa\_enb}}$	H2	Add Pull-up
PTXCLAV	53	txa_clav	J5	None
PTXCLK	82	txa_clk	H3	None
PTXPAR	50	txa_par	N3	None
PRXDATA[7:0]	57, 58, 59, 62, 63, 64, 65, 66	rxa_data[7:0]	E2, E3, E4, D3, D4, D5, C5, B5	None
PRXSOC	56	rxa_soc	H4	None
$\overline{\text{PRXEN}}$	55	$\overline{\text{rxa\_enb}}$	B6	Connect to $\overline{\text{enb}}$ of the PHY. Add pull-up.
PRXCLAV	54	rxa_clav	A6	None
PRXCLK	79	rxa_clk	C6	None
		phya_alm	C2	Pull-down
		phya_rx_led	D2	Pull-down
		phya_tx_led	H5	Pull-down
		txa_data[15:8]	N4, N5, M2, M3, M4, L2, L3, L4	No connect
		rxa_data[15:8]	G3, G4, G5, F1, F2, F3, F4, F5	Pull-down
		rxa_par	G2	Pull-down

Table 3 - UTOPIA Interface Pins for PHY & MT90500

2.4.2 PHY & MT90500 & SAR

In this configuration the MT90500 is interfaced to a PHY device via its Primary port. Another device is present on the Primary receive bus, such as an AAL5 SAR. In this setup it is likely that the PRXEN of the MT90500 and the enb of the PHY are both connected to the output enb of the AAL5 SAR. The MT90500 Primary port is operating in 8-bit ATM mode and the Secondary port is operating in 8-bit PHY mode.

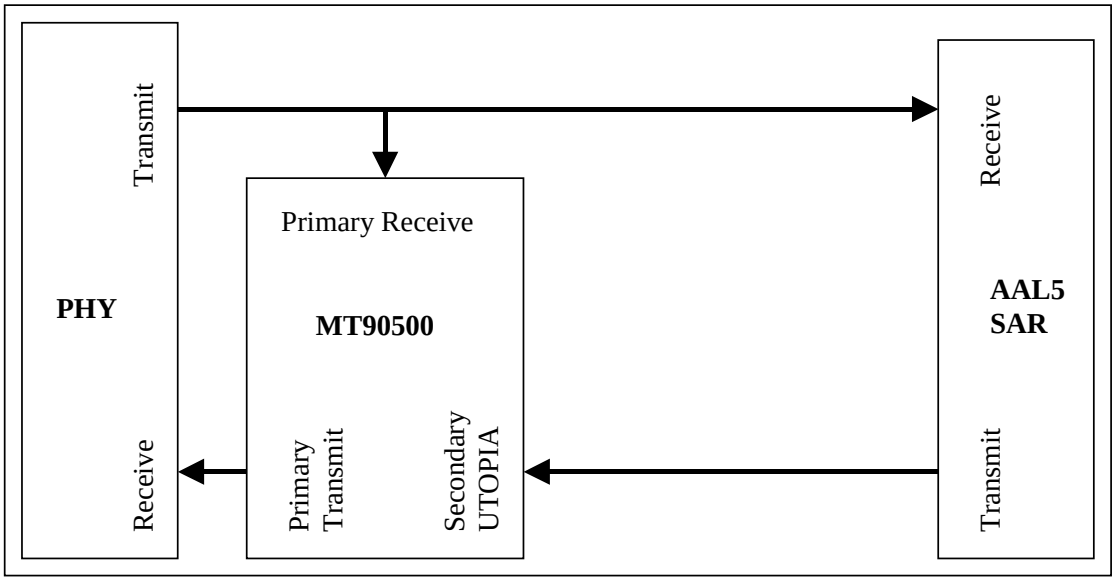

Figure 4 - PHY & MT90500 & PHY

The MT90503 will duplicate this operation by making a direct one-to-one connection between the MT90503 Port A and the PHY device, as well as making a direction one-to-one connection between MT90503 Port B and the AAL5 SAR. Port A will be configured in ATM mode and Port B will be configured in PHY mode. Cells may be received on Port B from the AAL5 SAR and transmitted out Port A towards the PHY. Cells may be received on Port A from the PHY and transmitted out Port B towards the AAL5 SAR. The MT90503 will use both Port A and Port B, configured for 8-bit, Level 1, ATM mode and PHY mode respectively.

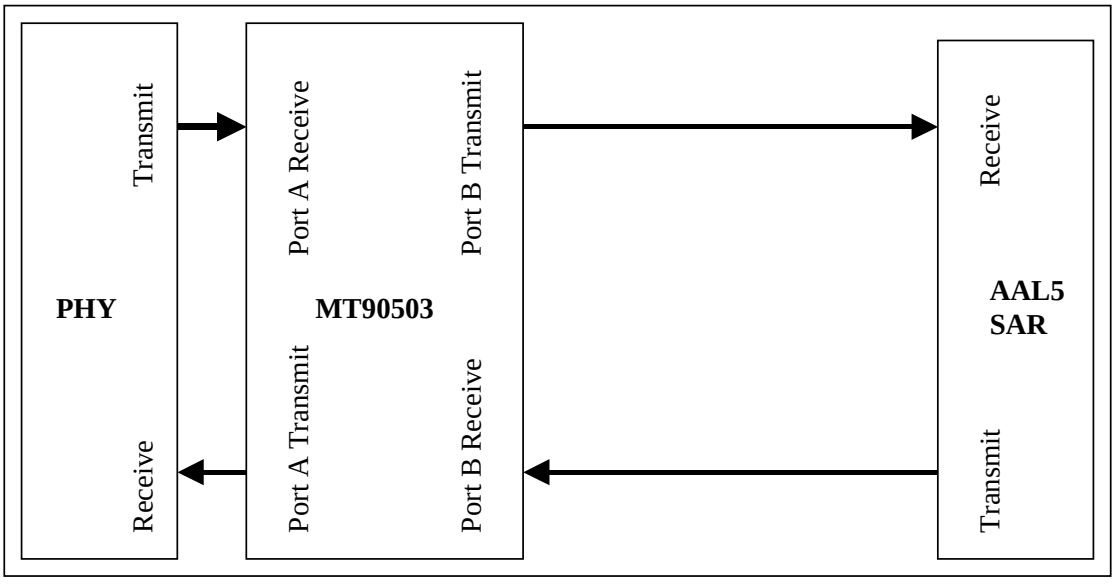

Figure 5 - PHY & MT90503 & SAR

In this configuration Table 3 and Table 4 will list the Primary receive port twice as the interface will be split into two sections for Port A and Port B as indicated above. One connection will be between the PHY and the MT90503 Port A receive, the other between the AAL5 SAR and the MT90503 Port B transmit. Port A's configuration is the same as shown in section 2.4.1 PHY & MT90500. Port B's configuration is listed below.

In this configuration MT90503 Port C is unused. See section 2.4.5 MT90503 Port C Unused for details.

MT90500 Pin Name	MT90500 Pin Number	MT90503 Equivalent Pin Name	MT90503 Equivalent Pin Number	Action
STXDATA[7:0]	70, 71, 72, 73, 74, 75, 76, 77	rxb_data[7:0]	T3, T4, R2, R3, R4, R5, P2, P3	None
STXSOC	69	rxb_soc	V2	None
$\overline{\text{STXEN}}$	68	$\overline{\text{rxb\_enb}}$	P4	None
STXCLAV	67	rxb_clav	P5	Add pull-down
STXCLK	85	rxb_clk	N1	None
PRXDATA[7:0]	57, 58, 59, 62, 63, 64, 65, 66	txb_data[7:0]	AA2, AA3, AA4, AA5, Y2, Y3, Y4, Y5	None
PRXSOC	56	txb_soc	AD2	None
$\overline{\text{PRXEN}}$	55	$\overline{\text{txb\_enb}}$	W2	None
PRXCLAV	54	txb_clav	W3	Add pull-down
PRXCLK	79	txb_clk	W4	None
		phyb_alm	AB5	Pull-down
		phyb_rx_led	W5	Pull-down
		phyb_tx_led	T5	Pull-down
		txb_data[13:8]	AC3, AC4, AG2, AB2, AB3, AB4	No connect
		txb_data[14]	AC2	No connect
		txb_data[15]	AD4	No connect
		txb_par	AD3	No connect
		rxb_data[11:8]	U3, U4, U5, T2	Pull-down
		rxb_data[15:12]	V4, V5, U1, U2	Pull-down
		rxb_par	V3	Pull-down

Table 4 - UTOPIA Interface Pins for PHY & MT90500 & SAR

2.4.3 PHY & MT90500 & MT90500 & SAR

In this configuration the MT90500 is interfaced to a PHY device via its Primary port. Two other devices are present on the Primary receive bus, another MT90500 and an AAL5 SAR. In this setup it is likely that the PRXEN of both the MT90500's and the enb of the PHY are connected to the output enb of the AAL5 SAR. The MT90500's are operating in 8-bit ATM mode.

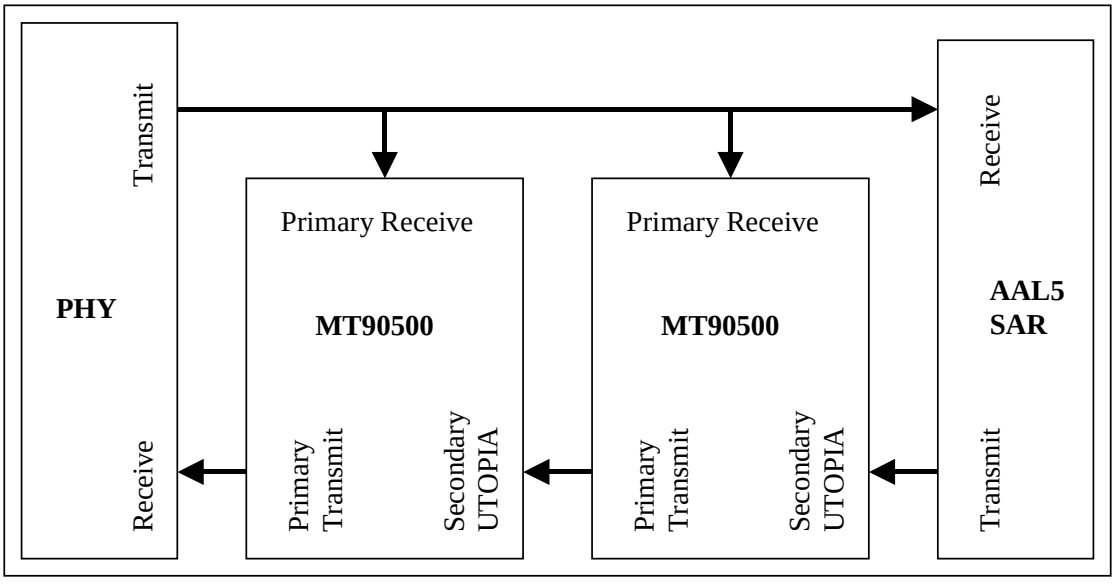

Figure 6 - PHY & MT90500 & MT90500 & SAR

Since the MT90503 has twice the capacity of the MT90500, a single MT90503 can replace both the MT90500 devices. The configuration is the same as listed in section 2.4.2 PHY & MT90500 & SAR.

In this configuration MT90503 Port C is unused. See section 2.4.5 MT90503 Port C Unused for details.

However, if a second MT90503 were used to replace the second MT90500, then the second MT90503's Port A would be connected to Port C of the first MT90503. Cells could be received on Port C of the first MT90503 and transmitted out Port B of the first MT90503 towards the AAL5 SAR or Port A of the first MT90503 towards the PHY. Cells received on either Port A or Port B of the first MT90503 could be sent out Port C of the first MT90503 towards Port A of the second MT90503 SAR.

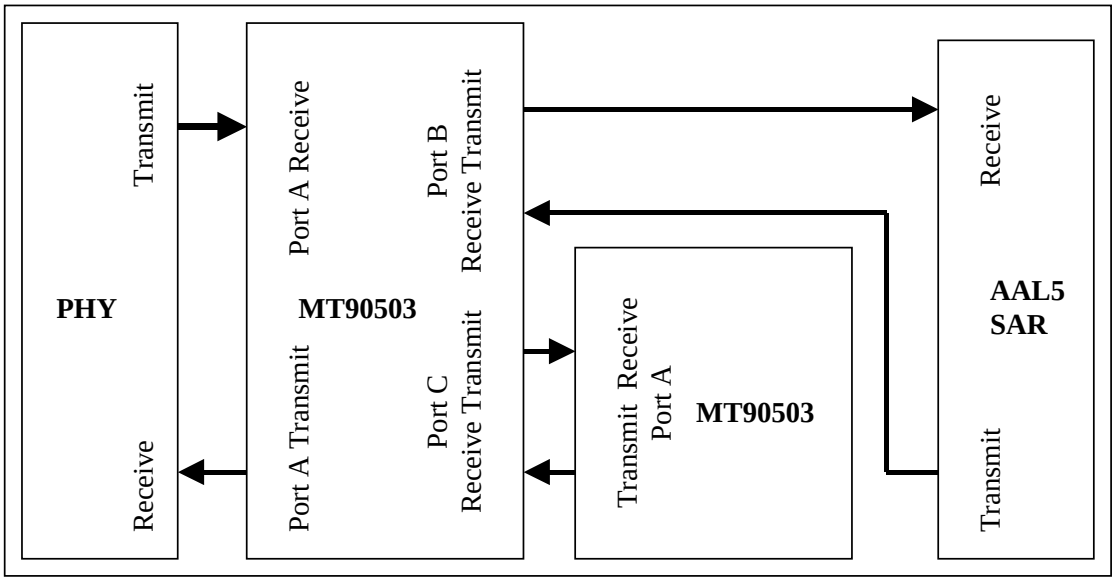

Figure 7 - PHY & MT90503 & MT90503 & SAR

2.4.4 MT90503 Port B Unused

If Port B of the MT90503 is not used, the table below shows the recommended pull-ups or pull-downs. On power-up or reset, Port B is configured for PHY mode of operation with its clocks as inputs. By default, the MT90503 API open chip function configures Port B for ATM mode of operation with the clocks as inputs.

MT90500 Pin Name	MT90500 Pin Number	MT90503 Equivalent Pin Name	MT90503 Equivalent Pin Number	Action
		txb_clk	W4	Pull-down
		rxb_clk	N1	Pull-down
		phyb_alm	AB5	Pull-down
		phyb_rx_led	W5	Pull-down
		phyb_tx_led	T5	Pull-down
		txb_clav	W2	Pull-down
		$\overline{\text{txb\_enb}}$	W3	No connect
		txb_soc	AD2	No connect
		txb_data[7:0]	AA2, AA3, AA4, AA5, Y2, Y3, Y4, Y5	No connect
		txb_data[13:8]	AC3, AC4, AG2, AB2, AB3, AB4	No connect
		txb_data[14]	AC2	No connect
		txb_data[15]	AD4	No connect
		txb_par	AD3	No connect
		rxb_clav	P4	Pull-down
		$\overline{\text{rxb\_enb}}$	P5	No connect
		rxb_soc	V2	Pull-down
		rxb_data[7:0]	T3, T4, R2, R3, R4, R5, P2, P3	Pull-down
		rxb_data[11:8]	U3, U4, U5,T2	Pull-down
		rxb_data[15:12]	V4, V5, U1, U2	Pull-down
		rxb_par	V3	Pull-down

Table 5 - UTOPIA Interface Pins for Port B Unused

2.4.5 MT90503 Port C Unused

If Port C of the MT90503 is not used, the table below shows the recommended pull-ups or pull-downs. On power-up or reset, Port C is configured for PHY mode of operation with its clocks as inputs. By default, the MT90503 API open chip function configures Port C for PHY mode of operation with its clocks as inputs.

MT90500 Pin Name	MT90500 Pin Number	MT90503 Equivalent Pin Name	MT90503 Equivalent Pin Number	Action
		txc_clk	AG10	Pull-down
		rxc_clk	AF7	Pull-down
		$\overline{\text{txc\_enb}}$	AE10	Pull-up
		txc_clav	AF10	No connect
		txc_soc	AH9	No connect
		txc_data[7:0]	AG9, AF9, AE9, AH8, AG8, AF8, AE8, AH7	No connect
		txc_par	AG7	No connect
		$\overline{\text{rxc\_enb}}$	AH6	Pull-up
		rxc_clav	AJ6	No connect
		rxc_soc	AG6	Pull-down
		rxc_data[7:0]	AF6, AE6, AF5, AF4, AE2, AE3, AF3, AD1	Pull-down
		rxc_par	AH10	Pull-down

Table 6 - UTOPIA Interface Pins for Port C Unused

2.5 Memory Interface

The MT90500 memory interface supports two chip selects/banks of either flowthrough or pipelined SSRAM. The data bus is 32-bits wide with 14 to 18 address lines. There is the option of using two 16-bit SSRAM chips instead of one 32-bit SSRAM chip through the second $\overline{CS}$ on each bank interface. By contrast, the MT90503 supports two independent memory interfaces. The control memory interface has two chip selects/banks while the data memory interface has four chip selects/banks. The MT90503 supports either flowthrough or pipelined SSRAM or ZBT-SRAM. The memory interface configurations are summarized below. Greater details on the MT90503 memory interface can be found in the Memory Overview section of its data sheet.

Feature	MT90500	MT90503
Number of memory interfaces	1 (mixed control and data)	2 (control and data)
Number of chip selects/banks	2 (each chip select is comprised of a high and low)	2 for control 4 for data
Maximum size per chip select/bank	1 MB	1 MB
Data bus (bits)	32	16
Address lines	14 - 18	16 - 19
flowthrough SSRAM support	yes	yes
pipelined SSRAM support	yes	yes
flowthrough ZBT support	no	yes
pipelined ZBT support	no	yes
output memory clock	yes	yes
input memory clock	no	yes

Table 7 - Memory Interface Configurations

Both the MT90500 and MT90503 have 3.3V memory interfaces which are 5V tolerant.

A typical MT90500 memory configuration would have two chip selects/banks, with the chip selects divided into lower (L) and upper (U) allowing two chips per bank. Each bank would use 18 address lines and a 32-bit data bus over the two chips. The two banks would have a total of 2 MB of SSRAM combined.

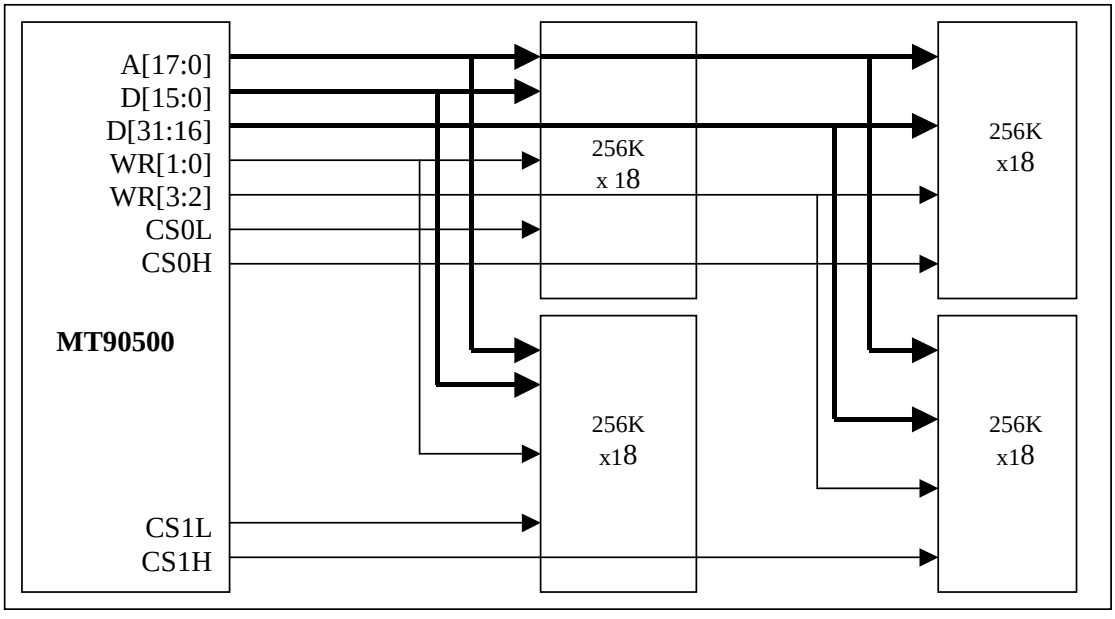

Figure 8 - Typical MT90500 Memory Configuration

The similar configuration for the MT90503 would have two chip selects/banks of control memory and two chip selects/banks of data memory. Each bank would have one chip using 18 address lines and a 16-bit data bus. The combined amount of memory used for both control and data would be 2 MB.

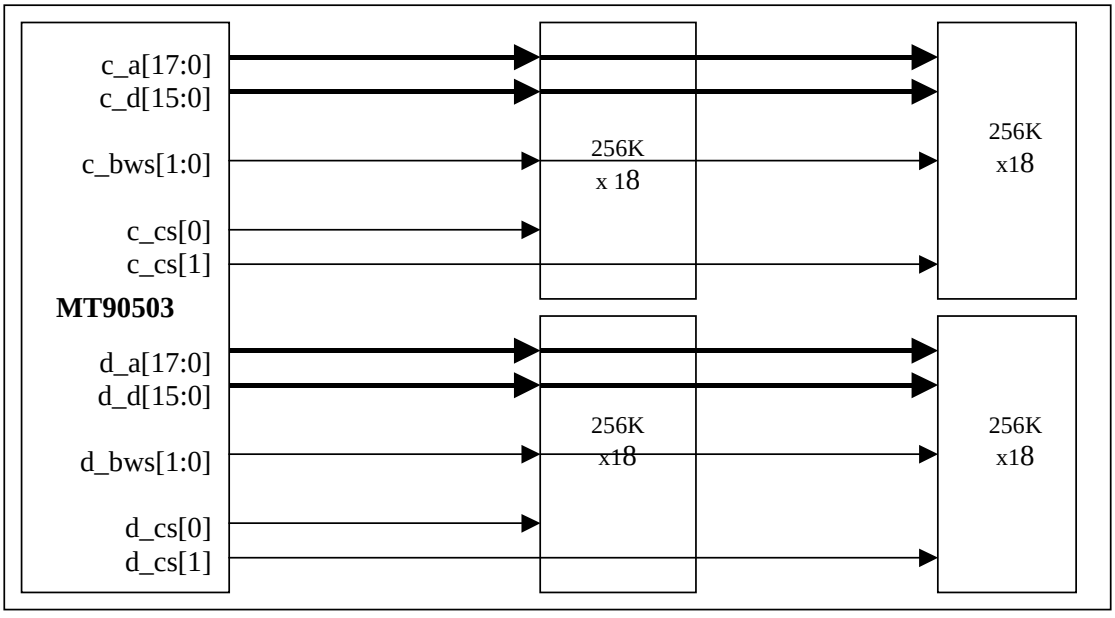

Figure 9 - Typical MT90503 Memory Configuration

The selection of how much SSRAM is required is different for the MT90503 than it is for the MT90500. The MT90500 had a common memory interface for both data and control memory, whereas the MT95003 has a separate memory interface for data and control memory.

The amount of control memory required is heavily dependent on the number of UTOPIA ports used and the size of the cell header look-up table. Details can be found in the MT90503 data sheet section Control Memory. A single SSRAM chip is able to support the maximum amount of control memory, which is 1 MB.

The amount of data memory required is dependent on the number of channels and the CDV of each channel. Assuming the each channel has the same circular buffer size (for the same CDV), the amount of data memory required for a non-multiframing application is roughly estimated by the formula:

Amount of memory [KB] = (number of channels x CDV [ms] x 4) / (125 x 1.024)

If a CDV of +/- 32 ms is desired for 1024 channels, then roughly 1024 KB of data memory would be required. This memory could be either two chips of 512 KB, as shown in Figure 9, or one chip of 1024 KB, using 19 data address lines.

Both the MT90500 and the MT90503 have output memory clocks, however the MT90503 also has an input memory clock which must be connected to the same clock as the external memory chips. Typically, the MT90503 output memory clock is fed to a clock distributor which in turn drives all the external memory chips and the input memory clock.

The three tables below show the typical memory configuration shown in Figure 8 and Figure 9. Table 8 shows common pins between both the MT90503 control and data memory. Table 9 shows the MT90503 control memory pins. Table 10 shows the MT90503 data memory pins. Some MT90500 memory interface pins will be listed twice as the MT90503 has separate pins for the control memory interface and the data memory interface. The MT90500 has only one set of pins for the mixed control and data memory interface.

MT90500 Pin Name	MT90500 Pin Number	MT90503 Equivalent Pin Name	MT90503 Equivalent Pin Number	Action
MEM_CLK	98	mem_clk_o	AD28	None
$\overline{\text{MEM\_WR[3:2]}}$	178, 179			None
$\overline{\text{MEM\_OE}}$	180			Pull-down $\overline{\text{OE}}$ pin on memory chip.
MEM_DAT[31:16]	166, 167, 168, 170, 171, 173, 174, 175, 153, 154, 155, 156, 158, 159, 162, 164			None
MEM_PAR[3:2]	165, 152			None
		mem_clk_i	AD29	Connect to memory clock
		mem_clk_positive_i	R26	Pull-up (PECL)
		mem_clk_negative_i	R25	Pull-down (PECL)
		mem_clk_positive_o	AF15	No connect
		mem_clk_negative_o	AE15	No connect

Table 8 - Memory Interface Pins for Data and Control Memory

MT90500 Pin Name	MT90500 Pin Number	MT90503 Equivalent Pin Name	MT90503 Equivalent Pin Number	Action
$\overline{\text{MEM\_CS0L}}$	147	cmem_$\overline{\text{cs}}$[0]	Y25	None
$\overline{\text{MEM\_CS0H}}$	176	cmem_$\overline{\text{cs}}$[1]	W26	None
$\overline{\text{MEM\_WR[1:0]}}$	149, 150	cmem_$\overline{\text{bws}}$[1:0]	Y26, Y27	None
MEM_ADD[17:0]	123, 122, 121, 118, 117, 116, 115, 103, 102, 99, 146, 144, 130, 128, 127, 126, 125, 124	cmem_a[17:0]	AG23, AH23, AE24, AF24, AG24, AH24, AJ24, AF25, AG25, AH25, AG26, AH26, AH27, AA26, AA27, AA28, W27, W28	None
MEM_DAT[15:0]	133, 134, 135, 136, 137, 138, 142, 143, 105, 106, 107, 108, 109, 112, 113, 114	cmem_d[15:0]	AG28, AF27, AE26, AE28, AD27, AC26, AC28, AB27, AF28, AE27, AD26, AC25, AC27, AB26, AB28, AA25	None
MEM_PAR[1:0]	131, 104	cmem_par[1:0]	AG27, AF26	None
		cmem_r/$\overline{\text{w}}$	Y28	Connect to memory chip if ZBT.

Table 9 - Memory Interface Pins for Control Memory

MT90500 Pin Name	MT90500 Pin Number	MT90503 Equivalent Pin Name	MT90503 Equivalent Pin Number	Action
$\overline{\text{MEM\_CS1L}}$	148	dmem_$\overline{\text{cs}}$[0]	U28	None
$\overline{\text{MEM\_CS1H}}$	177	dmem_$\overline{\text{cs}}$[1]	V27	None
$\overline{\text{MEM\_WR}}$[1:0]	149, 150	dmem_$\overline{\text{bws}}$[1:0]	U27, U26	None
MEM_ADD[17:0]	123, 122, 121, 118, 117, 116, 115, 103, 102, 99, 146, 144, 130, 128, 127, 126, 125, 124	dmem_a[17:0]	T28, K26, V28, U29, T27, T26, R28, K25, J27, J26, H28, H27, L25, H25, J28, G27, F25, F27, F29	None
MEM_DAT[15:0]	133, 134, 135, 136, 137, 138, 142, 143, 105, 106, 107, 108, 109, 112, 113, 114	dmem_d[15:0]	P25, N28, N26, M27, L28, L26, K28, K27, P28, P26, N29, N27, N25, M26, L27, M25	None
MEM_PAR[1:0]	131, 104	dmem_par[1:0]	P27, R27	None
		dmem_a[18]	T28	No connect
		dmem_$\overline{\text{rw}}$	U25	Connect to memory chip if ZBT.
		dmem_$\overline{\text{cs}}$[3:2]	V25, V26	No connect

Table 10 - Memory Interface Pins for Data Memory

2.6 TDM Interface

The MT90500 TDM interface (Serial TDM Bus) is compatible with ST-BUS, MVIP, H-MVIP, IDL and SCSA interfaces. The MT90503 TDM interface is H.100/H.110 compatible. The MT90500 and MT90503 TDM streams will both typically be configured to use an ST-BUS style 8.192 Mbps streams. For simplicity, the TDM configuration will consider 8.192 Mbps data streams with an input 8.192 MHz clock on CLKx1 and an input 8 kHz frame pulse on FSYNC. All other features will be assumed to be unused, unless otherwise noted. One note is that the MT90500 typically operates using a 61 ns wide frame pulse (for a 16 MHz clock), whereas the MT90503 operates using a 122 ns wide frame pulse (for an 8 MHz clock).

The MT90500 has a 5V interface. The MT90503 also has a 5V interface if the ct_vdd5[3:0] pins are connected to a 5V rail. The MT90503 has a 3.3V interface that is 5V compatible if the ct_vdd5[3:0] pins are connected to a 3.3V rail.

MT90500 Pin Name	MT90500 Pin Number	MT90503 Equivalent Pin Name	MT90503 Equivalent Pin Number	Action
ST[15:0]	25, 23, 22, 19, 18, 17, 15, 14, 12, 11, 10, 9, 8, 6, 5, 4	ct_d[15:0]	B20, B23, E19, C19, D23, D22, C22, D19, C24, B24, A24, E20, C21, B26, C23, D21	None
CLKx2PI	230			None
CLKx2NI	227			None
CLKx1	233	ct_c8_a	C28	None
FSYNC	232	ct_frame_a	C27	FSYNC is 61 ns wide ct_frame_a is 122 ns wide
IC	30			None
CORSIGA/CLKFAIL	32			None
CORSIGB/MC/FNXI	235	recov_a or ct_mc	E10 or F26	None
CORSIGC/MCTX/ SRTSENA	33	recov_b or mc_tx	D10 or B27	None
CORSIGD/MCRX/ SRTSDATA	34	recov_c or mc_rx	C10 or C25	None
CORSIGE/MCCLK	35	recov_d or mc_clock	B10 or E24	None
EX_8KA	83	ct_netref1	G25	None
SEC8K	234	ct_netref2	G26	None
REF8KCLK	226	recov_e	E11	None
PLLCLK	224	recov_f	D11	None
FREERUN	31			None
LOCx2	236			None
LOCx1	3			None
LSYNC	28			None
LOCSTo	26			None
LOCSTi	27			None
CLKx2/CLKx2PO	231			None
CLKx2NO	228			None

Table 11 - TDM Interface Pins

Table 12 provides recommendations for pins on the MT90503 that are not used. Some pins were listed in Table 11 and may be used to replace equivalent functions on the MT90500. Do not follow the recommendations in the following table for those pins that will be used to replace MT90500 functions.

MT90500 Pin Name	MT90500 Pin Number	MT90503 Equivalent Pin Name	MT90503 Equivalent Pin Number	Action
		ct_d[31:16]	B17, A17, E22, D17, B19, B18, E16, C18, C16, C20, E18, E17, D16, E21, B22, B21	Pull-down
		ct_netref1	G25	Pull-down
		ct_netref2	G26	Pull-down
		ct_c8_b	B28	Pull-down
		ct_frame_b	C26	Pull-down
		ct_frame_comp	E26	No connect
		ct_c2	D27	No connect
		ct_c4	D28	No connect
		ct_c16-	E23	No connect
		ct_c16+	D25	No connect
		ct_sclk	E27	No connect
		ct_sclkx2	E28	No connect
		ct_mc	F26	Pull-down
		mc_clock	E24	No connect
		mc_tx	B27	Tie high
		mc_rx	C25	No connect
		ct_vdd5_0	B25	5V or 3.3V
		ct_vdd5_1	D20	5V or 3.3V
		ct_vdd5_2	D18	5V or 3.3V
		ct_vdd5_3	C17	5V or 3.3V

Table 12 - TDM Interface Pins for MT90503 Unused

2.7 Miscellaneous

The following MT90500 pins were not covered in the previous sections. Their equivalent MT90503 pins are below.

The range of frequencies for the mclk_src of the MT90503 is 30 - 80 MHz.

The JTAG interfaces are both 3.3V, with the MT90503 having 5V tolerance.

The tristate pins are both 3.3V, with the MT90503 having 5V tolerance.

MT90500 Pin Name	MT90500 Pin Number	MT90503 Equivalent Pin Name	MT90503 Equivalent Pin Number	Action
MCLK	87	mclk_src	AF11	See above
$\overline{\text{RESET}}$	78	$\overline{\text{reset}}$	AE11	None
TMS	97	tms	AG4	None
TCK	93	tck	AH5	None
TDI	95	tdi	AG5	None
TDO	96	tdo	AH4	None
$\overline{\text{TRST}}$	94	trst	AE7	None
IC	89			None
$\overline{\text{TRISTATE}}$	90	$\overline{\text{global\_tri\_state}}$	AE17	None

Table 13 - Miscellaneous Pins

2.8 MT90503 Specific

The following section deals with the functions and pins that appear on the MT90503, but not on the MT90500. Some recovery pins were listed in Table 11 and may be used to replace equivalent functions on the MT90500. Do not follow the recommendations in the following table for those pins that will be used to replace MT90500 functions.

The Phase Lock Loop pins require special decoupling, as shown in the MT90503 data sheet figure PLL Pins Connections.

MT90500 Pin Name	MT90500 Pin Number	MT90503 Equivalent Pin Name	MT90503 Equivalent Pin Number	Action
		NC	A13, B3, B4, B7, B8, B9, B12, B13, B14, B15, B16, C3, C4, C7, C8, C9, C12, C13, C14, C15, D6, D7, D8, D9, D12, D13, D14, D15, D24, E5, E8, E9, E12, E13, E14, E15, E25, M5, M28, T25, W25, AB25, AE4, AE5, AE22, AE23, AE25, AF2, AF23	No connect
		recov_a	E10	Pull-down
		recov_b	D10	Pull-down
		recov_c	C10	Pull-down
		recov_d	B10	Pull-down
		recov_e	E11	Pull-down
		recov_f	D11	Pull-down
		recov_g	C11	Pull-down
		recov_h	B11	Pull-down
		pllvss_110	J25	See data sheet
		pllvdd_110	F28	See data sheet
		plllp2_110	H26	See data sheet
		pllagn_110	G28	See data sheet
		pllvss_300	AC5	See data sheet
		pllvdd_300	AG3	See data sheet
		proc_out	E7	No connect

Table 14 - MT90503 Specific Pins

3.0 Software Considerations

The MT90500 was supplied with a Programmer's Manual to aide in the software development process. The MT90500 data sheet contains detailed descriptions of its internal registers and the structures required to be written to external memory. The MT90503 data sheet contains the same information as the MT90500, such as the internal registers, external control structures and memory map. The information supplied in the data sheet allows a software device driver to be developed for the MT90503 using the same approach as taken with the MT90500.

Alternatively, an API is available for the MT90503 that is supplied free of charge. This API will form the core of a device driver for the MT90503. The API is written in C and is OS and processor independent. A typical application would use an embedded OS (e.g. VxWorks) running on a microprocessor (e.g. Motorola MPC860). The API would be located below the application layer software and above user supplied, board specific, functions such as low level reads and writes to the hardware. Complete details may be found in the MT90503 API User Guide.

A description of the system architecture, taken from the System Architecture section of the API User Guide, is presented below.

3.1 System Architecture

The API is structured such that the code is stateless. All state of the API is contained in user allocated memory. This memory is referred to as the instance structures of the chip. For every API function called by the user, one of the chip's instance structure pointers is provided as a parameter. This allows the API code to service multiple chips. The instance structure pointers may be stored by the user in an array, and indexed by chip number. When an API function is to be called, the appropriate pointer can then be retrieved from the list, via the chip's index, and passed to the function.

The system architecture of the API is described below for an embedded system in two different interrupt-handling methods: with and without deferred procedure calls. In the first case, a deferred procedure call is not used, and the API's ISR is called by the OS's ISR directly at the interrupt priority level. This architecture is depicted below. All blocks shaded in dark gray in the two figures are API code. All other blocks represent code provided by the user.

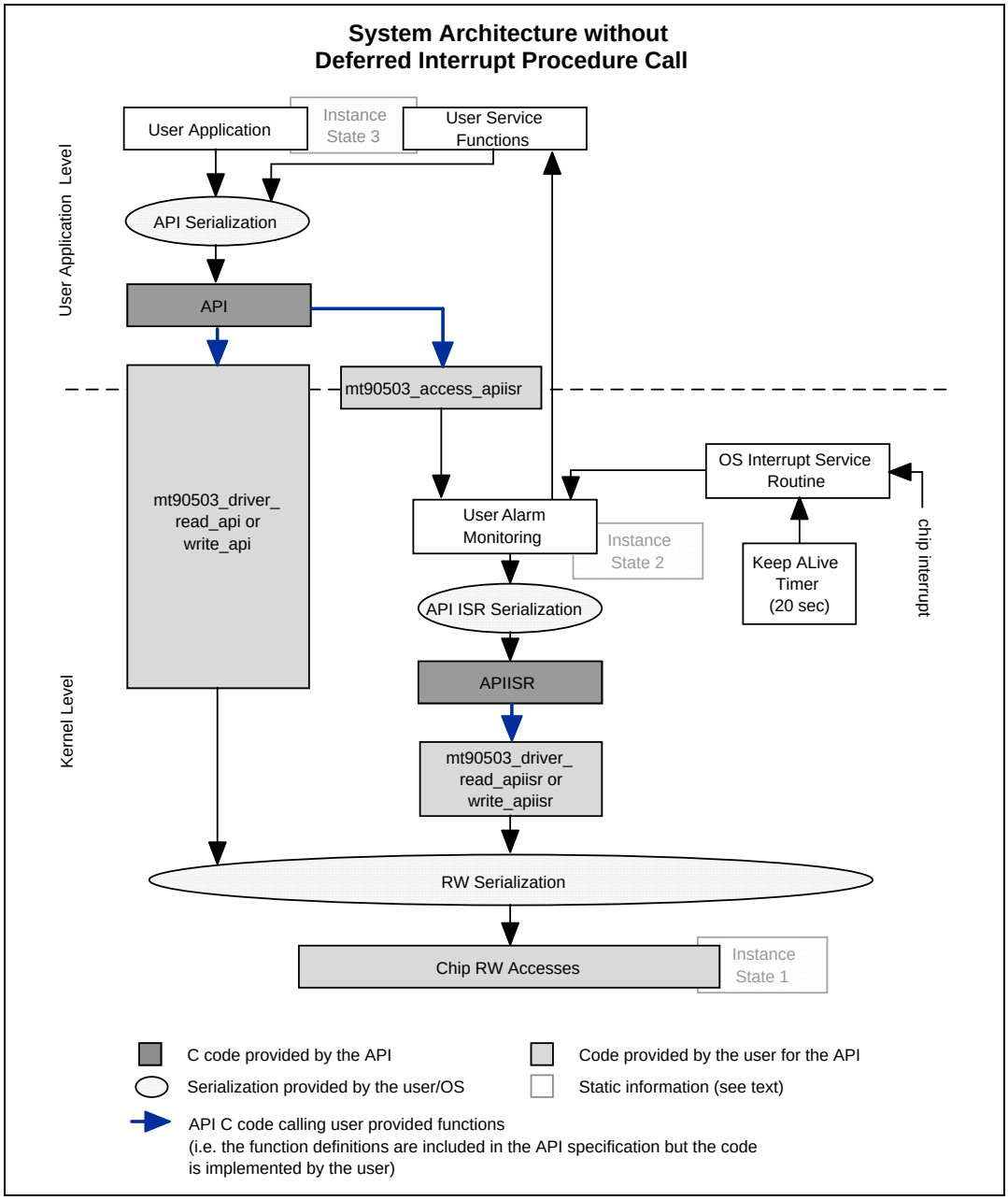

Figure 10 - System Architecture without Deferred Interrupt Procedure Call

Figure 11 depicts an architecture that uses deferred procedure calls. The OS's ISR simply defers the calling of the API's ISR to a later time, and at a lower interrupt priority level.

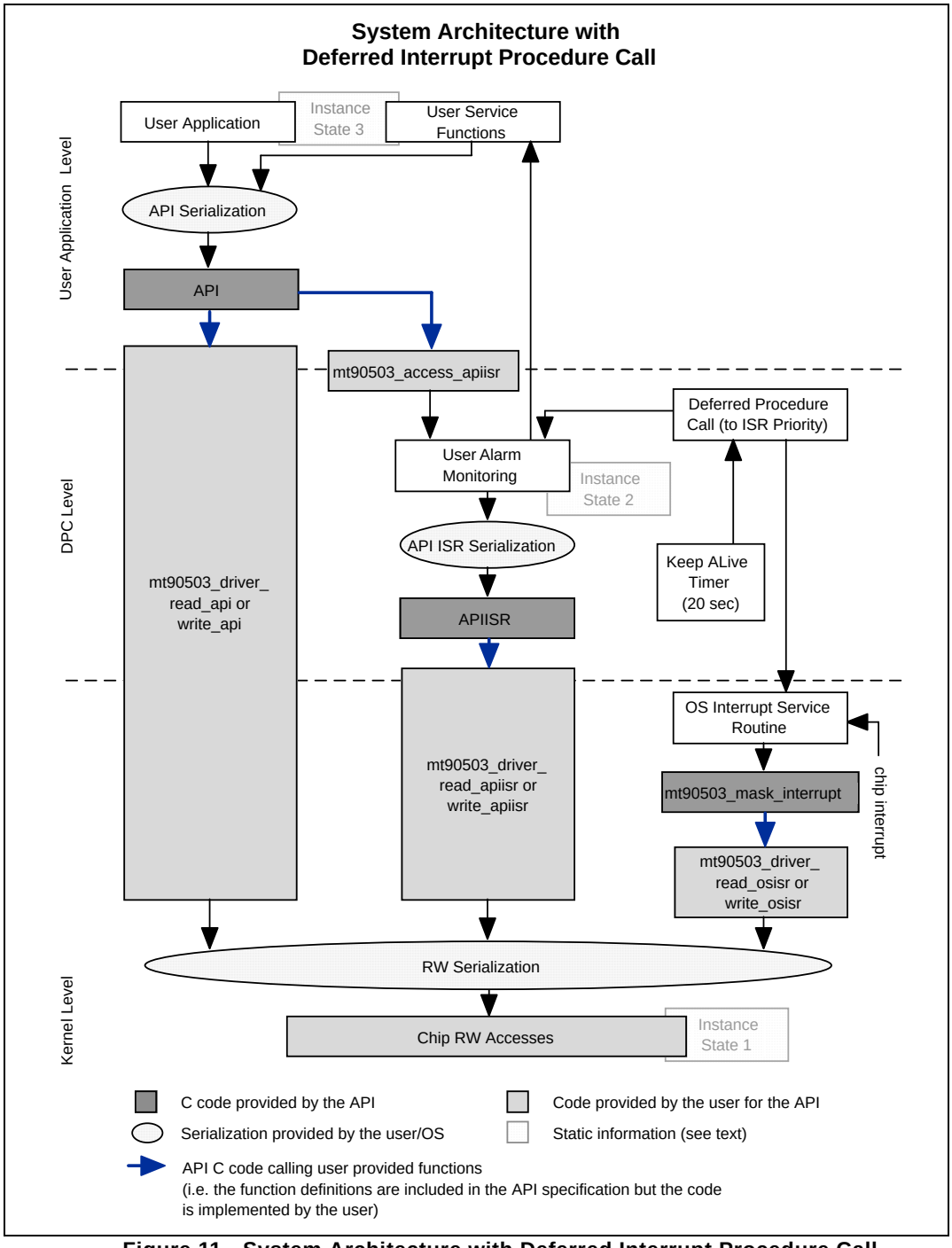

Figure 11 - System Architecture with Deferred Interrupt Procedure Call

In both architectures, an API serialization layer is needed to avoid a race condition between two threads, utilizing the same instance structure pointers, and attempting to call an API function. The serialization may be implemented in the form of a semaphore or mutex, for example. The serialization layer lies between the API and the user application (and is managed by the user).

Another serialization layer is needed for the APIISR code entity. The code entities are described below. Because the user is responsible for calling the APIISR within this entity when an interrupt is received, and because the API can itself call the APIISR, a race condition exists. The serialization performed at this level could be implemented via interrupt priority levels, for example.

Finally, the chip-level read and write accesses must be serialized as well. The serialization is necessary because a read/write access is split into several accesses to the CPU indirection registers of the chip. To insure that an access is completed correctly, these accesses to the indirection registers must be an atomic operation.

The API is contained in three different code entities, each of which may run at a different software/OS layer. The three sections correspond to the boxes in the figures labeled:

- API
- APIISR
- mt90503_mask_interrupt

The API code entity contains the majority of the functions that are called by the user, at a user priority level. These functions are not as fast as the APIISR function, and thus should not be serialized with interrupt execution. Because of this, and because the APIISR often runs at a higher priority level, the APIISR must be separated from the main API code.

The APIISR code entity contains the API's interrupt handling function. The function in this code entity is called by the OS's ISR upon receiving an interrupt from the hardware. It can also be called from the API code entity to access resources that the API and the ISR share. Thus, serialization between the OS's ISR's calls and the API's calls to this entity must be implemented by the user.

The API's ISR does not have to run at the same priority level as the OS's ISR. To do this, the interrupt signal line must be masked out in the MT90503 to be able to execute at a lower priority level (temporarily disabling the interrupt). The smallest code entity, mt90503_mask_interrupt, performs this task. It does so with only two accesses to the chip: one read and one write. The read is performed to query the state of the chip's interrupt register. This is necessary for systems that have multiple devices sharing an interrupt line. If the chip has flagged an interrupt, a write is performed to the chip to disable the interrupt pin. It executes very quickly, thus allowing other high-priority interrupts to be serviced immediately. Because this function has no access to the instance structures of the chip, it need not be serialized with any other part of code. This function masks out all interrupts for a period of 16ms. If the API's ISR has not completed within 16 ms of masking out the interrupts, another interrupt will be generated. If no interrupt is present, the function will return a status code that allows the user to avoid an unnecessary call to the APIISR.

Because the API and APIISR entities may lie in different OS priority levels, and because some OSs protect and separate kernel space memory from user space memory, the two code entities do not access the same memory. Each code entity needs a pointer to its own distinct block of memory. The API entity needs a pointer to a block of user space memory, and the APIISR entity needs a block of kernel space memory. Each portion can only access its own memory block.

As stated earlier, the API and APIISR entities share some information. Some API functions need to return information that is gathered by the APIISR and stored in its instance structure. Because the API does not have direct access to the APIISR's instance structure, the API is given access to the APIISR's information through a user supplied function, mt90503_access_apiisr. The function serves as a "messaging pipe" between the two entities. See the function description **mt90503_access_apiisr**.

As stated earlier, the API is structured to support multiple chips. Each chip instance requires its own pair of pointers to user allocated memory: the API instance structure and the APIISR instance structure. These pointers can be stored in an array, and indexed by chip number. When the OS enforces independent memory spaces two arrays must be kept: one in the user application's memory space for the API instance structures, and the other in the ISR's memory space for the APIISR structures. The two arrays are depicted in the figures above as "Instance State 3" and "Instance State 2", respectively.

The size of these memory blocks is determined by the API function mt90503_open_instance_size, described later in this document. The function is called for each chip, before initially configuring it. The function takes a chip configuration structure as a parameter and uses it to return the memory size required for the API and APIISR instance structures. See the function description **mt90503_open_instance_size**.

The read and write routines supplied by the user are used by the API functions to access all chips which the API code is servicing. The chip and its associated instance structure are configured via a call to the function mt90503_open. The function receives a chip configuration structure as a parameter. In this structure is the user_chip_number parameter that is intended to be the index of the chip being opened. Because every API and APIISR function receives a pointer to an instance structure as the first parameter, the chip number is available to all API functions. The only use of the user_chip_number by the API is to provide it as a parameter to the read/write functions. By associating a chip number to a particular chip, the correct device can be accessed in the user provided read/write routines. For example, chip number could be associated to a base address in the system. The user can then offset the provided address of a read/write routine and perform an access to the correct device. As illustrated in the two figures above, this information is easily stored as an array of chip specific information (e.g. base addresses) and can be indexed by the chip number. Note that the same chip number can be used to access system arrays kept by the user in different memory regions (e.g. user vs. kernel):

API instance structure pointer array (Instance State 3),

APIISR instance structure pointer array (Instance State 2),

Read/Write function chip info (Instance State 1).

The two figures above indicate that two or three versions of the same read and write functions must be supplied. These functions differ only in the layer of their entry point. The functions in the group mt90503_driver_read_api or write_api are accessible only from the user application space, the group mt90503_driver_read_apiisr or write_apiisr from the DPC priority level in kernel space, and the group mt90503_driver_read_osisr or write_osisr from the interrupt priority level in the kernel space. In the case where deferred procedure calls are not used, the third group is not needed.

The mt90503_interrupt_service_routine, located in the APIISR code entity, returns a vector of the interrupts that were serviced; it is the responsibility of the User Alarm Monitoring function to call any required user functions to continue the servicing in the user application. For example, if the user wanted to service data cells (AAL0) as soon as they are received, the rx_data_fifo_stale_time parameter would be set to the minimum desired delay value (e.g. 1ms) and the alarm_data_cell_fifo_int_conf parameter would be set to MT90503_INT_NO_TIMEOUT. (The two parameters are part of the chip configuration structure MT90503_CONF.) When a data cell arrives, the chip would assert an interrupt at most 1ms later. In response to the interrupt the OS ISR calls the API ISR, which services the interrupt and returns the vector indicating a data cell interrupt. The User Alarm Monitoring function then calls the user routine (in the user space) for data cells, which calls the API routine mt90503_receive_data_cell to obtain the cell.

Figure 12 depicts the system architecture used to perform the debugging of the API. The architecture is implemented on a Windows NT platform. Note that the API's ISR is located in the user space to facilitate debugging. Also important is the presence of a separate thread. This thread is dedicated to handling interrupts only. It waits for a flag from the OS's ISR indicating that an interrupt has been generated. Upon receiving the flag, the interrupt thread calls the API's ISR. The thread then performs appropriate actions based on the value of the event vector returned by the API's ISR.

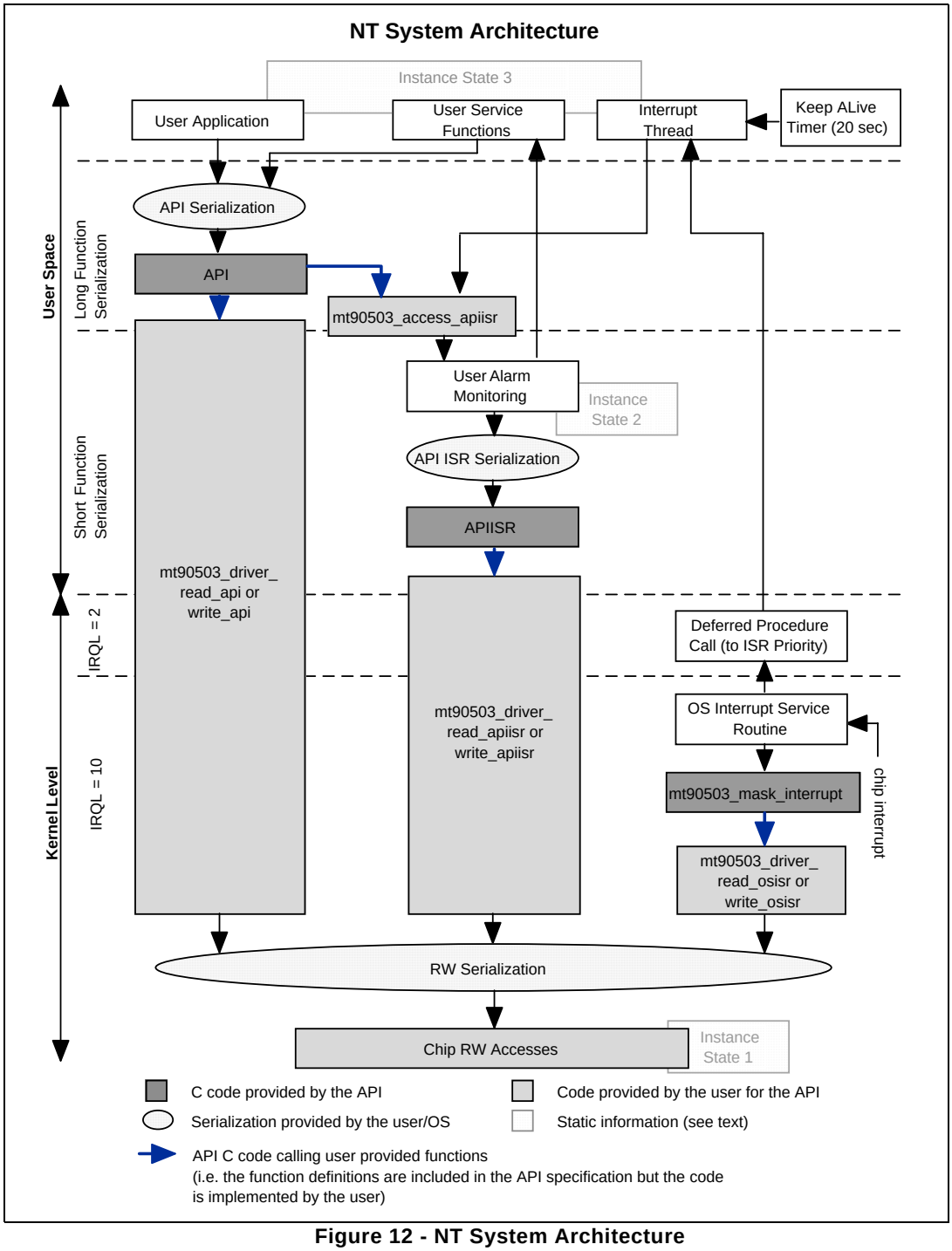

Figure 12 - NT System Architecture

The APIISR must be called at least every 20 seconds. If it is not, counters within the chip will not be updated in the API correctly, causing the API to fall out of syncronization with the chip, which can lead to system to failure. In a system where the interrupt line of the chip is routed to a CPU, APIISR code insures that the APIISR will be called at least at the required frequency. In the case where the interrupt line is not physically routed, a keep-alive timer is needed by the system, as illustrated in the system architecture figures above. The timer insures the APIISR is called at least every 20 seconds. Although calling the APIISR every 20 seconds is enough to keep the chip running correctly, VC and chip statistics counters also have to be kept up to date via calls to the **mt90503_poll_vc_stats** and the **mt90503_poll_chip_stats** functions. If such calls are not done frequently enough, incorrect statistics may be present in the API structures.

4.0 Summary

As the recommended replacement for the MT90500 AAL1 SAR, the MT90503 is able to supply identical or nearly identical operation as the MT90500 without having to spend a significant amount of time re-designing an existing board. The hardware replacement, as discussed in the Hardware Considerations section, provides straight-forward guidelines for configuring the MT90503 to provide the same interface functionality as the MT90500. The software replacement, as discussed in the Software Considerations section, provides two options when moving to the MT90503. The first option is to use the same approach as taken with the MT90500 to develop device driver software. The second option is to use the API provided to develop the necessary device driver. The MT90503 has many features and capabilities that were not discussed in this document. These capabilities are detailed in the MT90503 product brief and data sheet, available from the Zarlink Semiconductor web site.

For more information about all Zarlink products
visit our Web Site at

www.zarlink.com

TECHNICAL DOCUMENTATION - NOT FOR RESALE