APPLICATION NOTE

Implementation of SDRAM on SAMA5D2x Devices

Atmel | SMART SAMA5D2 Series

Scope

The Atmel® | SMART SAMA5D2 Series is a high-performance, power-efficient embedded MPU based on the ARM® Cortex®-A5 processor.

The SAMA5D21, SAMA5D22, SAMA5D23, SAMA5D24, SAMA5D26, SAMA5D27 and SAMA5D28 eMPUs feature one multi-port DDR controller that supports 32-bit DDR3(L)-SDRAM, 32-bit LPDDR3-SDRAM, 32-bit DDR2-SDRAM, 32-bit LPDDR2-SDRAM and 32-bit LPDDR1-SDRAM memories. These memories are called SDRAM in this document.

SAMA5D2x embeds a pad calibration feature that performs bus impedance adaptation, improving signal integrity. This leads to a reduction of overshoots, of Electromagnetic Interference (EMI), of power consumption on I/Os and eliminates the need of serial resistors on data lines.

This application note helps the developer design a system using external memory.

Reference Documents

Type	Title	Literature No.
Datasheet	SAMA5D2 Series Datasheet	11267

Table of Contents

1. Multi-port DDR Controller Overview

The Multi-port DDR-SDRAM Controller (MPDDRC) is a multi-port memory controller. It comprises eight slave AHB interfaces. All simultaneous accesses (eight independent AHB ports) are interleaved to maximize memory bandwidth and minimize transaction latency due to DDR-SDRAM protocol.

The MPDDRC extends the memory capabilities of a chip by providing the interface to the external 16-bit or 32-bit DDR-SDRAM device. The page size supports ranges from 2048 to 16384 rows and from 256 to 4096 columns. It supports dword (64-bit), word (32-bit), half-word (16-bit), and byte (8-bit) accesses.

The MPDDRC supports a read or write burst length of eight locations. This enables the command and address bus to anticipate the next command, thus reducing the latency imposed by the DDR-SDRAM protocol and improving the DDR-SDRAM bandwidth. Moreover, MPDDRC keeps track of the active row in each bank, thus maximizing DDRSDRAM performance, e.g., the application may be placed in one bank and data in other banks. To optimize performance, avoid accessing different rows in the same bank. The MPDDRC supports a CAS latency of 2, 3 or 6 and optimizes the read access depending on the frequency.

Self-refresh, Power-down and Deep Power-down modes minimize the consumption of the DDR-SDRAM device.

OCD (Off-chip Driver) and ODT (On-die Termination) modes are not supported.

The MPDDRC supports DDR3-SDRAM and DDR3L-SDRAM devices with DLL disabled, in DLL Off mode. In this mode, according to JEDEC standard, the maximum clock frequency is 125 MHz. However, check with memory suppliers for higher speed support. DDR3-SDRAM supports high capacity, 1 Gbit and more, and enables to reduce power consumption with a 1.5V supply (DDR3-SDRAM) or a 1.35V supply (DDR3L-SDRAM). The DLL Off mode sets the CAS Read Latency (CRL) and the CAS Write Latency (CWL) to 6. The latency is automatically set by the controller.

The MPDDRC I/Os are powered by VDDIODDR. For DDR2-SDRAM and LPDDR1-SDRAM, VDDIODDR is set to 1.8V nominal; for DDR3-SDRAM, VDDIODDR is set to 1.50V nominal; for DDR3L-SDRAM, VDDIODDR is set to 1.35V nominal; for LPDDR2-SDRAM and LPDDR3-SDRAM, VDDIODDR is set to 1.2V nominal.

The DDR Chip Select enables to have 512 Mbytes of SDRAM, from address 0x2000 0000 to address 0x4000 0000 for standard accesses, and from address 0x4000 0000 to address 0x6000 0000 for AES encrypted accesses.

2. Multi-Port DDR Controller Signals Definition

The MPDDRC manages 4-bank and 8-bank SDRAM devices. The signals generated by the controller are defined in Table 2-1.

Table 2-1. SDRAM Controller Signals

Signal Name	Function	Type	Active Level	Description
DDR_RESETN	DDR3 Asynchronous Reset	Input	–	Prevent illegal commands and/or unwanted states. When Self-refresh mode is used, should be tied to VDDIODDR using 100 KΩ, pull-up.
DDR_VREF	Reference Voltage	Input	–	Used by the input buffers of the DDR2 memories as well as the DDR2 controller to determine logic levels. VREF is specified to be ½ the power supply voltage and is created using a voltage divider constructed from two 1.5 KΩ, 1% tolerance resistors.
DDR_CAL	LPDDR2 Calibration Reference	Input	–	Used to calibrate I/O. See Calibration section for more details.
DDR_CK, DDR_CKN	DDR2 Differential Clock	Output	–	Differential clock signals that feed the SDRAM device. All other signals take those two signals as a reference.
DDR_CKE	DDR2 Clock Enable	Output	High	Acts as an inhibit signal to the DDR device. DDR_CKE remains high during valid DDR2 access (Read, Write, Prech). This signal goes low when the device is in Power-down mode or in Self-refresh mode; a self-refresh command can be issued by the controller (refer to the DDR2 controller Self-refresh mode).
DDR_CS	DDR2 Controller Chip Select	Output	Low	When the Chip Select (DDR_CS) is low, the command input is valid. When it is high, the commands are ignored but the operation continues.
DDR_BA[2..0]	Bank Select	Output	Low	Select the bank to address when a command is input. Read/write or precharge is applied to the bank selected by DDR_BA0, DDR_BA1, or DDR_BA2.
DDR_WE	DDR2 Write Enable	Output	Low	The Row Address Strobe (DDR_RAS) and the Column Address Strobe (DDR_CAS) will assert to indicate that the corresponding address is present on the bus. The conjunction with Write Enable (DDR_WE) and chip select (SDCS), at the rising edge of the clock (DDR_CK) or the falling edge of the #clock (#DDR_CK), determines the DDR2 operation.
DDR_RAS - DDR_CAS	Row and Column Signal	Output	Low	
DDR_A[13..0]	DDR2 Address Bus	Output	–	SDRAM controller address lines, respectively bounded to [A0:A13] on the microcontroller.
DDR_D[31..0]	DDR2 Data Bus	I/O/-PD	–	SDRAM controller data lines, respectively bounded to [DDR_D31:DDR_D0] on the microcontroller.
DDR_DQS[3..0], DDR_DQSN[3..0]	Differential Data Strobe	I/O/-PD	–	DDR_DQS[0..3]: Data Strobe. The data is sampled on DDR_DQS edges. DDR_DQSN[0..3]: Negative Data Strobe, for LPDDR2-SDRAM. DQSN must be connected to DDR_VREF for DDR2 memories.

Table 2-1. SDRAM Controller Signals (Continued)

Signal Name	Function	Type	Active Level	Description
DDR_DQM[3..0]	Write Data Mask	Output	–	Data is accessed in 32 bits by means of DDR_DQM[3..0], which are respectively the highest to lowest mask bits for the DDR2 data on the bus.

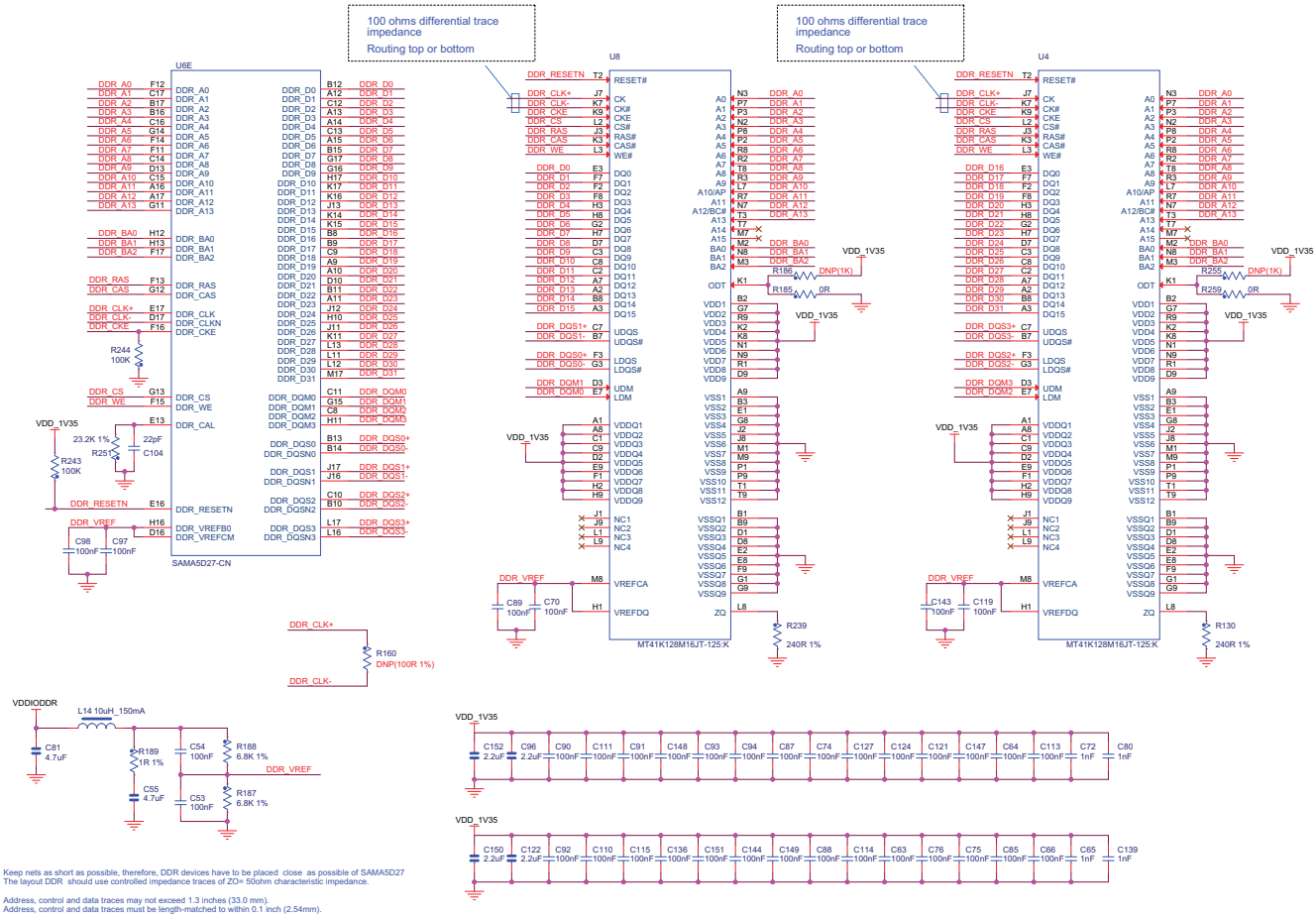
3. SDRAM Connection on SAMA5D2x

The user interface to configure the MPDDR controller is mapped at address 0xF000 C000.

Each memory device must use sufficient decoupling to provide an efficient filtering on the power supply rails.

3.1 32-bit Using 2x16-bit DDR3(L)-SDRAM Implementation

3.1.1 Hardware Configuration

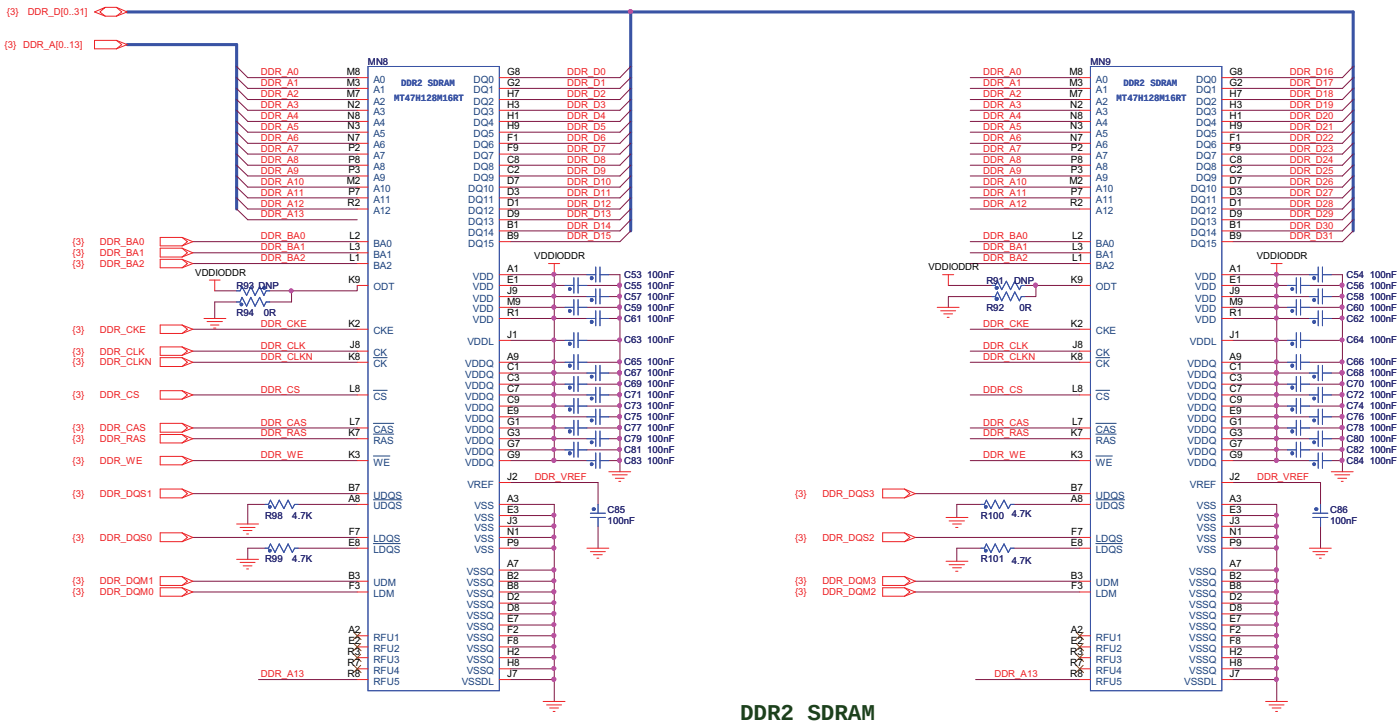


3.1.2 Software Configuration

Refer to Section 7.1 “DDR3(L)-SDRAM Initialization” for information on the DDR3 initialization sequence.

3.2 32-bit Using 2x16-bit DDR2-SDRAM Implementation

3.2.1 Hardware Configuration



3.2.2 Software Configuration

Refer to Section 7.1 “DDR3(L)-SDRAM Initialization” for information on the DDR2 initialization sequence.

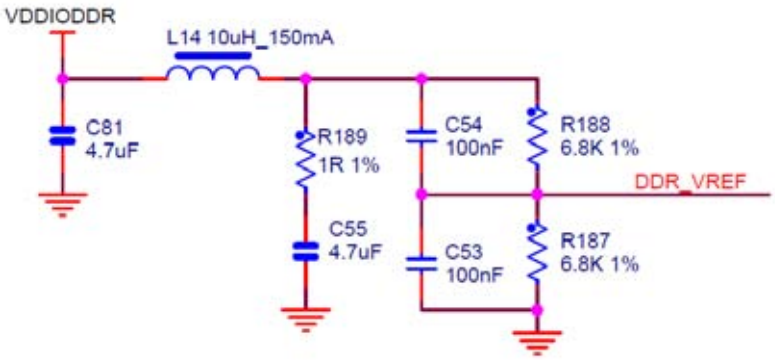
3.3 DDR2-SDRAM and DDR3(L)-SDRAM VREF Signal Considerations

DDR_VREF, which is half the interface voltage, is provided by a voltage divider of VDDIODDR.

- 0.9V for 1.80V typical VDDIODDR for DDR2 Interface I/O lines
- 0.68V for 1.35V typical VDDIODDR for DDR3L Interface I/O lines
- 0.75V for 1.50V typical VDDIODDR for DDR3 interface I/O lines

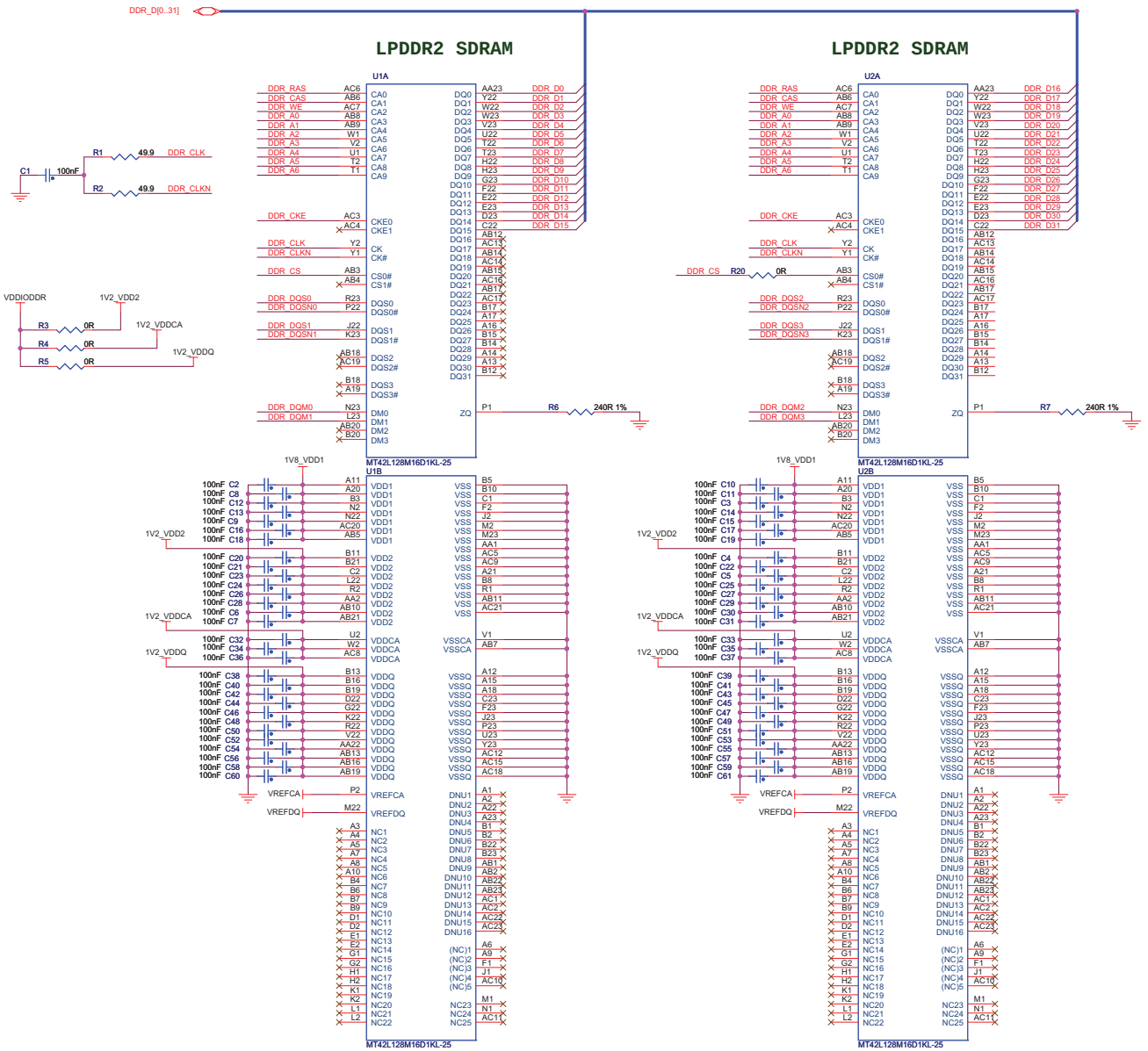
DDR_VREF is not a high current supply, but it is important to keep it as noiseless as possible with minimal inductance. DQSN[3:0] must be connected to DDR_VREF for DDR2-SDRAM and DDR3(L)-SDRAM memories.

Figure 3-1. Example for DDR3L-SDRAM



3.4 32-bit Using 2x16-bit LPDDR2-SDRAM Implementation

3.4.1 Hardware Configuration



3.4.2 Software Configuration

Refer to Section 7.4 “LPDDR2-SDRAM Initialization” for information on the LPDDR2 initialization sequence.

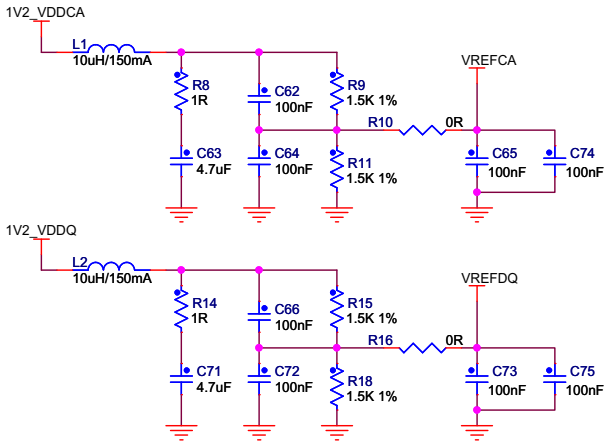
3.5 LPDDR1-SDRAM, LPDDR2-SDRAM and LPDDR3-SDRAM VREF Signal Considerations

DDR_VREF, which is half the interface voltage, is provided by a voltage divider of VDDIODDR.

- 0.9V for 1.80V typical VDDIODDR for LPDDR Interface I/O lines
- 0.6V for 1.20V typical VDDIODDR for LPDDR2 / LPDDR3 Interface I/O lines

DDR_VREF is not a high current supply, but it is important to keep it as noiseless as possible with minimal inductance.

To reduce noise, two VREF pins are needed for LPDDR2-SDRAM: VREFCA and VREFDQ.



4. Layout and Design Constraints

4.1 General Considerations

This section provides routing guidelines for layout and design of a printed circuit board using high-speed memories. The signal integrity rules for high-speed interfaces need to be considered. In fact, it is highly recommended that the board design be simulated to determine optimum layout for signal integrity and quality. Keep in mind that this document can only highlight the most important issues that should be considered when designing a board with high-speed memories. The designer has to take into account the corresponding information (specification, design guidelines, etc.) contained in the documentation for each interface that is to be implemented on board.

The length difference between the data lane and the CK signal should not exceed 400 mils.

In each data lane (e.g. lane0 includes DQ[7:0]/DM0/DQS0), the length difference between each signal and the respective DQS/DQSn signal should not exceed 100 mils.

The following rules should also be observed:

- 3W rule: keep the distance between two adjacent traces larger than 3 times the trace width.
- Serpentine routing rule: when a design requires equal-length traces between the source and multiple loads, some traces can be bent in order to match the trace lengths. However, improper trace bending affects signal integrity and propagation delay. To minimize crosstalk, ensure that $S \geq 3 \times H$, where S is the spacing between the parallel sections and H is the height of the signal trace above the reference plane.

4.2 DDR2/LPDDR1 Bus Interface Controller

Bus signals can be split in three groups:

1. Differential Clock source, VREF middle voltage point for DDR2 reference voltage
   - This first and most critical signal group is set up by the CK, NCK signals and the VREF reference voltage only.
2. Bus, Strobe and Mask signals
   - Data bus signals
   - DQS signals
   - DQM signals
3. Address, Control signals
   - Address bus signals (DDR_Ax)
   - DDR_BAx signals (Bank select signals)
   - CKE signal, RAS, CAS, NWE and NCS control signals

4.2.1 Group 1 Signals

Group 1 signals are the most critical, and should be routed first. The clock is driven in Differential mode. Two traces should be planned to drive those signals to DDR2 packages. The clock traces have the same impedance, and therefore:

- Both traces must be routed on the outer layer.
- Both traces must be parallel.
- Clock traces must use only two vias/traces.

Voltage reference VREF can be impacted by noise. Those voltage reference traces versus other traces must be kept away from noisy digital traces (in three dimensions: above and below layers and on both sides): maintain a 15–20 mils clearance from other nets.

This net should be larger than other traces (like a small local voltage plane), and located on the layer closest to the ground layer.

4.2.2 Group 2 Signals

Route Group 2 signals by keeping the propagation delay equal as first constraint:

- Route each data group (DQ + DQS + DM) on the same layer to match propagation delays and minimize skew between these signals.
- Between signals DQ and DQM, keep a minimum space of 3 widths; between DQS (potentially impacted by noise) and DQ/DQM, keep a minimum space of 4 widths.

4.2.3 Group 3 Signals

Group 3 signals should be routed by minimizing crosstalk with [DQ, DQS, DQM] ↔ [Addresses, CTRL Bus]: maintain a gap between the two groups and do not interlace them.

Table 4-1. Example of PCB Stackup

Layer	Type	Description
Layer 1 (Top)	Signal	Differential and critical signals: oscillators, quartz, clock, DDR_VREF, DDR_CLK / DDR_NCLK, etc. address/data buses
Layer 2	GND	GND
Layer 3	Signal	Address/data buses, non-critical signal
Layer 4	Signal	Address/data buses, non-critical signal
Layer 5	Power plane	Non-critical signal
Layer 6 (Bottom)	Signal	Differential and critical signals: oscillators, quartz, clock, DDR_VREF, DDR_CLK / DDR_NCLK, etc. address/data buses

4.3 EBI Trace Routing Guidelines

4.3.1 Topology About the EBI Bus

Bus impedance: maintain an impedance of 50Ω to ±10%.

4.3.2 Placement

Place the highest-speed/highest-current components as far from the I/O connections as possible.

4.3.3 Bypassing Capacitors

Keep all surface traces that run between the pads of the decoupling capacitors and their vias as short and wide as possible. Use as small a body size for a decoupling capacitor as you can afford, and minimize the length of all connections from the capacitor pads to the power and ground planes.

4.3.4 Trace Length

Keep the time delay of stubs less than 20% of the rise time of the fastest signals.

Route the shortest path between MPU and resistor networks dedicated to the memories.

4.3.5 Trace Spacing

For microstrip or stripline transmission lines, keep the spacing between adjacent signal paths at least twice the line width.

Keep all traces at least five line widths from the edge of the board.

4.3.6 Vias

Use vias as large in diameter as practical when routing to power or ground planes.

4.3.7 Ground Plane(s)

Follow the return path of each signal and keep the width of the return path under each signal path at least as wide, and preferably at least three times as wide, as the signal trace.

To avoid EMIs, avoid routing switching signals across splits or openings in ground planes. Routing around them is preferable even if it results in longer paths.

4.3.8 Power Plane(s)

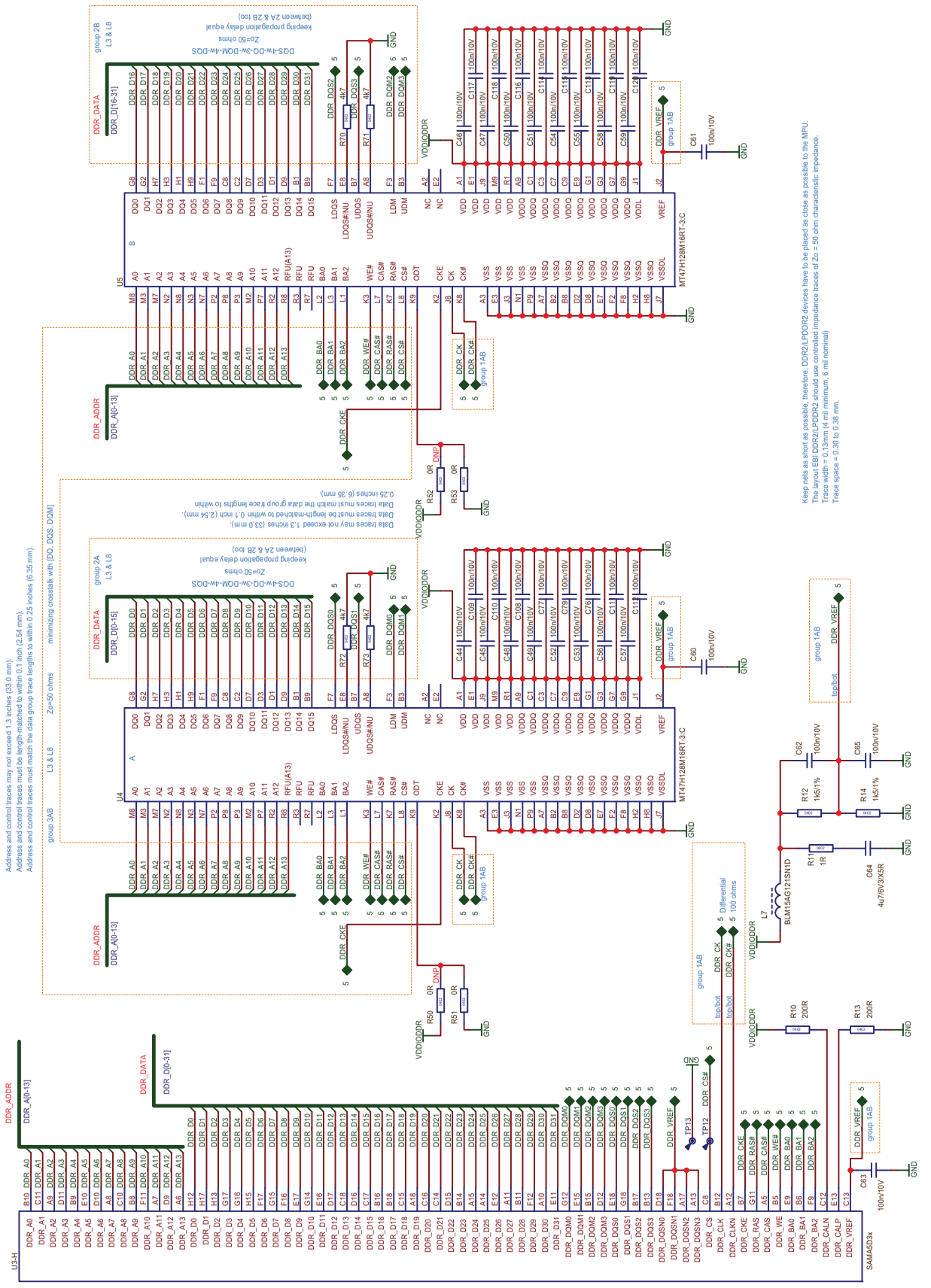
- Minimize the loop inductance between the power and ground paths.
- Allocate power and ground planes on adjacent layers with as thin a dielectric as possible.
- Route the power and ground planes as close as possible to the surface where the decoupling capacitors are mounted.
- Supply voltages must be composed of planes only, not traces. Short connections (≈ 8 mils) are commonly used to attach vias to planes. Any connections required from supply voltages to vias for device pins or decoupling capacitors should be as short and as wide as possible to minimize trace impedance (20 mils trace width).

4.3.9 General Considerations for High-Speed Differential Interfaces

The following is a list of suggestions for designing with high-speed differential signals.

- Use controlled impedance PCB traces that match the specified differential impedance.
- Keep the trace lengths of the differential signal pairs as short as possible.
- The differential signal pair traces should be trace-length matched and the maximum trace-length mismatch should not exceed the specified values. Match each differential pair per segment.
- Maintain parallelism and symmetry between differential signals with the trace spacing needed to achieve the specified differential impedance.
- Maintain maximum possible separation between the differential pairs, any high-speed clocks/periodic signals (CMOS/TTL) and any connector leaving the PCB (such as I/O connectors, control and signal headers, or power connectors).
- Route differential signals on the signal layer nearest the ground plane using a minimum of vias and corners. This will reduce signal reflections and impedance changes. Use GND stitching vias when changing layers.
- Route CMOS/TTL and differential signals on a different layer(s), which should be isolated by the power and ground planes.
- Avoid tight bends. When it becomes necessary to turn 90°, use two 45° turns or an arc instead of making a single 90° turn.
- Do not route traces under crystals, crystal oscillators, clock synthesizers, magnetic devices or ICs that use, and/or generate, clocks.
- Stubs on differential signals should be avoided due to the fact that stubs will cause signal reflections and affect signal quality.
- Keep the length of high-speed clock and periodic signal traces that run parallel to high-speed signal lines at a minimum to avoid crosstalk. Based on EMI testing experience, the minimum suggested spacing to clock signals is 50 mils.
- Use a minimum of 20 mils spacing between the differential signal pairs and other signal traces for optimal signal quality. This helps to prevent crosstalk.
- Route all traces over continuous planes (VCC or GND), avoiding to cross splits or openings in those planes.

Figure 4-1. SAMA5D2x-CM Memory Schematic



5. LPDDR2-SDRAM and LPDDR3-SDRAM Power-up and Power-off Considerations

5.1 Power-up Sequence

A specific sequence must be used to power up the LPDDR2-SDRAM or LPDDR3-SDRAM device. This procedure is mandatory. Power-up and initialization by means other than those specified will result in undefined operation.

Refer to the LPDDR2-SDRAM or LPDDR3-SDRAM datasheet for full details and timings.

5.2 Power-off Sequence

A specific sequence must be used to power off the LPDDR2-SDRAM or LPDDR3-SDRAM device. This procedure is mandatory. Power-off by means other than those specified will result in uncontrolled power-off.

The VDDIODDR power fail must be handled at system level (IRQ or FIQ). When this event occurs, the LPDDR2-LPDDR3 power-off sequence is to be applied using the LPDDR2_LPDDR3_PWOFF bit (bit 3 in MPDDRC_LPR) before the VDDIODDR power-off.

Uncontrolled power-off sequence can be applied only up to 400 times in the life of a LPDDR2-SDRAM or LPDDR3-SDRAM device.

Refer to the LPDDR2-SDRAM or LPDDR3-SDRAM datasheet for full details and timings.

6. Calibration Considerations

SAMA5D2x embeds a pad calibration feature that performs bus impedance adaptation, improving signal integrity. This leads to:

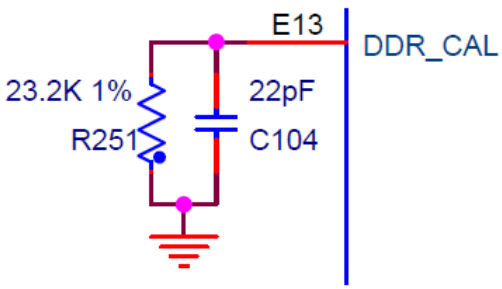
- reduction of overshoots,
- reduction of Electromagnetic Interference (EMI),
- reduction of power consumption on I/Os,
- eliminating the need of a serial resistor on data lines.

6.1 Hardware

Calibration requires connecting a resistor on DDR_CAL to GND. The resistor value depends on the SDRAM type:

- 24 KΩ for LPDDR2/LPDDR3
- 23 KΩ for DDR3L
- 22 KΩ for DDR3
- 21 KΩ for DDR2/LPDDR1

Figure 6-1. Example for DDR3L-SDRAM



6.2 Software

6.2.1 DDR3(L)-SDRAM, DDR2-SDRAM and LPDDR1-SDRAM Calibration

DDR3, DDR2 or LPDDR1 software calibration is to be done only once and requires the following steps:

1. Set field RDIV in the MPDDRC_IO_CALIBR register, according to the board impedance.
2. Calculate value TZQIO using the formula TZQIO = (DDRCLK × 20 ns) + 1.
3. Set the TZQIO time in MPDDRC_IO_CALIBR.
4. Activate calibration by setting the 5th bit in the High Speed Register (0xFFFFEA24).

6.2.2 LPDDR2-SDRAM and LPDDR3-SDRAM Calibration

LPDDR2 and LPDDR3 software calibration is to be done only once and requires the following steps:

1. Set field RDIV in the MPDDRC_IO_CALIBR register, according to the board impedance.
2. Calculate value TZQIO using the formula TZQIO = (DDRCLK × 20 ns) + 1.
3. Set the TZQIO time in MPDDRC_IO_CALIBR.
4. Program Short Calibration Time with field ZQCS in LPDDR2_TIM_CAL according to the LPDDR2(3)-SDRAM datasheet.
5. Calculate the calibration pulse over Process Voltage Temperature (PVT) according to the refresh rate, the temperature and voltage expected change, the temperature and voltage sensitivities defined in the LPDDR2(3)-SDRAM datasheet, using the formula: COUNT_CAL = ZQCorrection / ((TSens × Tdriftrate) + (VSens × Vdriftrate)).
6. Set the value in field COUNT_CAL in the LPDDR23_CAL_MR4 register.

For example, if TSens = 0.75%/°C, VSens = 0.2%/mV, Tdriftrate = 1°C/sec and driftrate = 15mV/sec, then the interval between ZQCS commands is calculated as 1.5 / ((0.75 × 1) + (0.2 × 15)) = 0.4 sec.

This LPDDR2-SDRAM device requires a calibration every 0.4s.

The value to be loaded depends on the average time between REFRESH commands, t_{REF}.

For an LPDDR2-SDRAM with a time between refresh of 7.8 µs, the value of the Calibration Timer Count bit is programmed ($0.4/7.8 \times 10^{-6}$) = 0xC852.

7. Multi-port DDR Controller Configuration

7.1 DDR3(L)-SDRAM Initialization

The DDR3(L)-SDRAM initialization sequence is described in Section “DDR3-SDRAM/DDR3L-SDRAM Initialization” of the SAMA5D2 Series datasheet. For an example of initialization, see “DDR3(L)-SDRAM Initialization Code Example”.

7.2 LPDDR3-SDRAM Initialization

The LPDDR3-SDRAM initialization sequence is described in Section “Low-power DDR3-SDRAM Initialization” of the SAMA5D2 Series datasheet.

7.3 DDR2-SDRAM Initialization

The DDR2-SDRAM initialization sequence is described in Section “DDR2-SDRAM Initialization” of the SAMA5D2 Series datasheet. For an example of initialization, see “DDR2-SDRAM Initialization Code Example”.

7.4 LPDDR2-SDRAM Initialization

The LPDDR2-SDRAM initialization sequence is described in Section “Low-power DDR2-SDRAM Initialization” of the SAMA5D2 Series datasheet. For an example of initialization, see “LPDDR2-SDRAM Initialization Code Example”.

7.5 LPDDR1-SDRAM Initialization

The LPDDR1-SDRAM initialization sequence is described in Section “Low-power DDR1-SDRAM Initialization” of the SAMA5D2 Series datasheet.

7.6 Micron® MT41K128M16JT DDR3L-SDRAM (MPDDRC Configuration Example)

Micron MT41K128M16JT devices are 256 MB DDR3L-SDRAM devices arranged as 16 Mbit × 16 × 8 banks with a CAS latency of 5 at 166 MHz. These devices are featured on the SAMA5D2-XULT.

Table 7-1 gives the delay in ns extracted from the DDR3L-SDRAM datasheet, the corresponding number of cycles at 166 MHz, and the field to program these values accordingly in a system running at 498 MHz for Processor Clock and 166 MHz for System Clock.

Table 7-1. MPDDRC Configuration Example with Micron MT41K128M16JT

Description	System Configuration	Value in Micron Datasheet	Number of Cycles at 166 MHz	Value in SAMA5D2 Datasheet	Register	Bit or Field	Register or Field Value
System							
PLL Frequency	996 MHz	–	–	–	CKGR_PLLAR	–	0x20523F01
Processor / Bus Clock	498 / 166 MHz	–	–	–	PMC_MCKR	–	0x00001302
System Clock	DDR clock enable	–	–	–	PMC_SCER	–	0x00000005
DDR3L Device							
MPDDRC Configuration Register(2)	–	–	–	–	MPDDRC_CR	–	0x00D0055D
Number of Columns	–	10	–	10	MPDDRC_CR	NC	0x1
Number of Rows	–	14	–	14	MPDDRC_CR	NR	0x3<<2
CAS Latency	–	5	5	5 cycles	MPDDRC_CR	CAS	0x5<<4

Table 7-1. MPDDRC Configuration Example with Micron MT41K128M16JT (Continued)

Description	System Configuration	Value in Micron Datasheet	Number of Cycles at 166 MHz	Value in SAMA5D2 Datasheet	Register	Bit or Field	Register or Field Value
Reset DLL	–	–	–	Disable	MPDDRC_CR	DLL	0x0<<7
Drive Strength (DDR2 only)	–	Normal	–	Normal	MPDDRC_CR	DIC_DS	0x1<<8
DLL OFF mode	–	–	–	DLL OFF	MPDDRC_CR	DIS_DLL	0x1<<9
Calibration (LPDD2/LPDDR3-SDRAM only)	–	–	–	Disabled	MPDDRC_CR	ZQ	0x0<<10
Off-Chip Driver	–	–	–	(1)	MPDDRC_CR	OCD	0x0<<12
Mask Data is shared	–	–	–	Not shared	MPDDRC_CR	DMQS	0x0<<16
Enable Read Measure	–	–	–	Disabled	MPDDRC_CR	ENRDM	0x0<<17
Number of banks	–	8	–	8	MPDDRC_CR	NB	0x1<<20
Not DQS (DDR2 only)	–	–	–	Disabled	MPDDRC_CR	NDQS	0x0<<21
Type of decoding	–	–	–	interleaved	MPDDRC_CR	DECOD	0x1<<22
Unaligned access allowed	–	–	–	Enabled	MPDDRC_CR	UNAL	0x1<<23
MPDDRC I/O Calibration Register(2)	–	–	–	–	MPDDRC_IO_CALIBR	–	0x00876504
Resistor Divider	–	–	–	RZQ_60_RZQ_57_RZQ_55_RZQ_52Ω	MPDDRC_IO_CALIBR	RDIV	0x4
Calibration Enable	–	–	–	–	MPDDRC_IO_CALIBR	EN_CALIB	0x0<<4
IO calibration	–	–	–	–	MPDDRC_IO_CALIBR	TZQIO	0x65<<8
Number of transistor P	–	–	–	–	MPDDRC_IO_CALIBR	CALCODEP	0x7<<16
Number of transistor N	–	–	–	–	MPDDRC_IO_CALIBR	CALCODEN	0x8<<20
MPDDRC Read Datapath Register(2)	–	–	–	–	MPDDRC_RD_DATA_PATH	–	0x00000002
Shift Sampling	–	–	–	2 cycles	MPDDRC_RD_DATA_PATH	SHIFT_SAMPLING	0x2
MPDDRC Timing Parameter 0 Register(2)	–	–	–	–	MPDDRC_TPR0		0x44439425
ACTIVATE to PRECHARGE Time (delay)	–	28 ns	5	–	MPDDRC_TPR0	TRAS	0x5
ACTIVATE to READ/WRITE Time (delay)	–	11 ns	2	–	MPDDRC_TPR0	TRCD	0x2<<4
Last DATA-IN to PRECHARGE Time (delay)	–	4 CK in DLL off mode	4	–	MPDDRC_TPR0	TWR	0x4<<8
REFRESH to ACTIVATE Time (delay)	–	49 ns	9	–	MPDDRC_TPR0	TRC	0x9<<12
PRECHARGE to ACTIVATE Time (delay)	–	14 ns	3	–	MPDDRC_TPR0	TRP	0x3<<16
ACTIVE BankA to ACTIVE BankB (delay)	–	Max of 7.5 ns or 4 CK	4	–	MPDDRC_TPR0	TRRD	0x4<<20
Internal Write to Read Delay	–	Max of 7.5 ns or 4 CK	4	–	MPDDRC_TPR0	TWTR	0x4<<24

Table 7-1. MPDDRC Configuration Example with Micron MT41K128M16JT (Continued)

Description	System Configuration	Value in Micron Datasheet	Number of Cycles at 166 MHz	Value in SAMA5D2 Datasheet	Register	Bit or Field	Register or Field Value
Load Mode Register Command to ACTIVE or REFRESH Command (delay)	–	Max of 7.5 ns or 4 CK	4	–	MPDDRC_TPR0	TMRD	0x4<<28
MPDDRC Timing Parameter 1 Register	–	–	–	–	MPDDRC_TPR1	–	0x03001D1B
Row Cycle Delay	–	160 ns	27	–	MPDDRC_TPR1	TRFC	0x1B
Exit Self-Refresh Delay to Non-Read Command	–	TRFC + 10 ns	29	–	MPDDRC_TPR1	TXSNR	0x1D<<8
Exit Self Refresh Delay to Read Command (not used in DDR3(L))	–	0 cycles	0	–	MPDDRC_TPR1	TXSRD	0x0<<16
Exit Power-down Delay to First Command	–	Max of 6 ns or 3 CK	3	–	MPDDRC_TPR1	TXP	0x3<<24
MPDDRC Timing Parameter 2 Register(2)	–	–	–	–	MPDDRC_TPR2		0x00074000
Read to Precharge	–	Max of 7.5 ns or 4 CK	4	–	MPDDRC_TPR2	TRTP	0x4<<12
Four Active Windows	–	40 ns	7	–	MPDDRC_TPR2	TFAW	0x7<<16
MPDDRC Memory Device Register	–	–	–	–	MPDDRC_MD		0x00000004
Memory Device	DDR2-SDRAM	–	–	–	MPDDRC_MD	MD	0x4
Data Bus Width	32 bits	–	–	–	MPDDRC_MD	DBW	0x0<<4
MPDDRC Refresh Timer Register - Timer Count	166 MHz	7.8 µs	–	–	MPDDRC_RTR	COUNT	0x510

Notes: 1. OCD is not supported, but it is a mandatory step in the DDR2 initialization phase.
2. Any bit or field of this register not listed in the table must remain unchanged.

Appendix A. DDR3(L)-SDRAM Initialization Code Example

This appendix provides an example of the DDR3L initialization code, associated with the different steps of the DDR3L-SDRAM initialization sequence. All defines are given in mpddrc.h or ddr.h.

```
int ddr3_sdram_initialize(unsigned int base_address,
     unsigned int ram_address,
     struct ddramc_register *ddramc_config)
{
  unsigned int ba_offset;

  /* Compute BA[] offset according to CR configuration */
  ba_offset = (ddramc_config->cr & AT91C_DDRC2_NC) + 9;
  if (!(ddramc_config->cr & AT91C_DDRC2_DECOD_INTERLEAVED))
    ba_offset += ((ddramc_config->cr & AT91C_DDRC2_NR) >> 2) + 11;

  ba_offset += (ddramc_config->mdr & AT91C_DDRC2_DBW) ? 1 : 2;

  dbg_very_loud(" ba_offset = %x ...\n", ba_offset);

  /*
   * Step 1: Program the memory device type in the MPDDRC Memory Device Register
   */
  write_ddramc(base_address, HDDRSDRC2_MDR, ddramc_config->mdr);

  /*
   * Step 2: Program features of the DDR3-SDRAM device in the MPDDRC
   * Configuration Register and in the MPDDRC Timing Parameter 0 Register
   * /MPDDRC Timing Parameter 1 Register
   */
  write_ddramc(base_address, HDDRSDRC2_CR, ddramc_config->cr);

  write_ddramc(base_address, HDDRSDRC2_T0PR, ddramc_config->t0pr);
  write_ddramc(base_address, HDDRSDRC2_T1PR, ddramc_config->t1pr);
  write_ddramc(base_address, HDDRSDRC2_T2PR, ddramc_config->t2pr);

  /*
   * Step 3: A NOP command is issued to the DDR3-SRAM.
   * Program the NOP command in the MPDDRC Mode Register (MPDDRC_MR).
   * The application must write a one to the MODE field in the MPDDRC_MR
   * Perform a write access to any DDR3-SDRAM address to acknowledge this command.
   * The clock which drive the DDR3-SDRAM device are now enabled.
   */
  write_ddramc(base_address, HDDRSDRC2_MR, AT91C_DDRC2_MODE_NOP_CMD);
  *((unsigned volatile int *)ram_address) = 0;

  /*
   * Step 4: A pause of at least 500us must be observed before a single toggle.
```

```
 */
 udelay(500);

/*
 * Step 5: A NOP command is issued to the DDR3-SDRAM
 * Program the NOP command in the MPDDRC_MR.
 * The application must write a one to the MODE field in the MPDDRC_MR.
 * Perform a write access to any DDR3-SDRAM address to acknowledge this command.
 * CKE is now driven high.
 */
write_ddramc(base_address, HDDRSDRC2_MR, AT91C_DDRC2_MODE_NOP_CMD);
*((unsigned volatile int *)ram_address) = 0;

/*
 * Step 6: An Extended Mode Register Set (EMRS2) cycle is issued to choose
 * between commercial or high temperature operations. The application must
 * write a five to the MODE field in the MPDDRC_MR and perform a write
 * access to the DDR3-SDRAM to acknowledge this command.
 * The write address must be chosen so that signal BA[2] is set to 0,
 * BA[1] is set to 1 and signal BA[0] is set to 0.
 */
write_ddramc(base_address, HDDRSDRC2_MR, AT91C_DDRC2_MODE_EXT_LMR_CMD);
*((unsigned int *)(ram_address + (0x2 << ba_offset))) = 0;

/*
 * Step 7: An Extended Mode Register Set (EMRS3) cycle is issued to set
 * the Extended Mode Register to 0. The application must write a five
 * to the MODE field in the MPDDRC_MR and perform a write access to the
 * DDR3-SDRAM to acknowledge this command. The write address must be
 * chosen so that signal BA[2] is set to 0, BA[1] is set to 1 and signal
 * BA[0] is set to 1.
 */
write_ddramc(base_address, HDDRSDRC2_MR, AT91C_DDRC2_MODE_EXT_LMR_CMD);
*((unsigned int *)(ram_address + (0x3 << ba_offset))) = 0;

/*
 * Step 8: An Extended Mode Register Set (EMRS1) cycle is issued to
 * disable and to program O.D.S. (Output Driver Strength).
 * The application must write a five to the MODE field in the MPDDRC_MR
 * and perform a write access to the DDR3-SDRAM to acknowledge this command.
 * The write address must be chosen so that signal BA[2:1] is set to 0
 * and signal BA[0] is set to 1.
 */
write_ddramc(base_address, HDDRSDRC2_MR, AT91C_DDRC2_MODE_EXT_LMR_CMD);
*((unsigned int *)(ram_address + (0x1 << ba_offset))) = 0;

/*
```

```
 * Step 9: Write a one to the DLL bit (enable DLL reset) in the MPDDRC
 * Configuration Register (MPDDRC_CR)
/*
 * Step 10: A Mode Register Set (MRS) cycle is issued to reset DLL.
 * The application must write a three to the MODE field in the MPDDRC_MR
 * and perform a write access to the DDR3-SDRAM to acknowledge this command.
 * The write address must be chosen so that signals BA[2:0] are set to 0
 */
write_ddramc(base_address, HDDRSDRC2_MR, AT91C_DDRC2_MODE_LMR_CMD);
*((unsigned int *)ram_address) = 0;

udelay(50);

/*
 * Step 11: A Calibration command (MRS) is issued to calibrate RTT and
 * RON values for the Process Voltage Temperature (PVT).
 * The application must write a six to the MODE field in the MPDDRC_MR
 * and perform a write access to the DDR3-SDRAM to acknowledge this command.
 * The write address must be chosen so that signals BA[2:0] are set to 0.
 */
write_ddramc(base_address, HDDRSDRC2_MR, AT91C_DDRC2_MODE_DEEP_CMD);
*((unsigned int *)ram_address) = 0;

/*
 * Step 12: A Normal Mode command is provided.
 * Program the Normal mode in the MPDDRC_MR and perform a write access
 * to any DDR3-SDRAM address to acknowledge this command.
 */
write_ddramc(base_address, HDDRSDRC2_MR, AT91C_DDRC2_MODE_NORMAL_CMD);
*((unsigned int *)ram_address) = 0;

/*
 * Step 13: Perform a write access to any DDR3-SDRAM address.
 */
*((unsigned int *)ram_address) = 0;

/*
 * Step 14: Write the refresh rate into the COUNT field in the MPDDRC
 * Refresh Timer Register (MPDDRC_RTR):
 * refresh rate = delay between refresh cycles.
 * The DDR3-SDRAM device requires a refresh every 7.81 us.
 */
write_ddramc(base_address, HDDRSDRC2_RTR, ddramc_config->rtr);

return 0;
```

Appendix B. DDR2-SDRAM Initialization Code Example

This appendix provides an example of the DDR2 initialization code, associated with the different steps of the DDR2-SDRAM initialization sequence.

```
//*----------------------------------------------------------------------------
//* \fn    ddram_init
//* \brief Initialization of the DDR Controller
//*----------------------------------------------------------------------------
int ddram_init(unsigned int ddram_controller_address, unsigned int ddram_address, struct
SDdramConfig *ddram_config)
{
   volatile unsigned int i;
   unsigned int cr = 0;

   // Initialization Step 1: Program the memory device type
   // Configure the DDR controller
   write_ddramc(ddram_controller_address, HMPDDRC_MDR, ddram_config->ddramc_mdr);
   // Program the DDR Controller
   write_ddramc(ddram_controller_address, HMPDDRC_CR, ddram_config->ddramc_cr);

   // Initialization Step 2: assume timings for 7.5 ns min clock period
   write_ddramc(ddram_controller_address, HMPDDRC_T0PR, ddram_config->ddramc_t0pr);
   // pSDDRC->HMPDDRC_T1PR
   write_ddramc(ddram_controller_address, HMPDDRC_T1PR, ddram_config->ddramc_t1pr);
   // pSDDRC->HMPDDRC_T2PR
   write_ddramc(ddram_controller_address, HMPDDRC_T2PR, ddram_config->ddramc_t2pr);

   // Initialization Step 3: NOP command -> allow to enable clk
   write_ddramc(ddram_controller_address, HMPDDRC_MR, AT91C_DDRC2_MODE_NOP_CMD);
   *((unsigned volatile int*) ddram_address) = 0;
   // Initialization Step 3 (must wait 200 µs) (6 core cycles per iteration, core is at 536 MHz:
   // min 17,733 loops)
   for (i = 0; i < 17800; i++) {
   asm("    nop");
   }

   // Initialization Step 4: A NOP command is issued to the DDR2-SDRAM
   // NOP command -> allow to enable cke
   write_ddramc(ddram_controller_address, HMPDDRC_MR, AT91C_DDRC2_MODE_NOP_CMD);
   *((unsigned volatile int*) ddram_address) = 0;
   // wait 400 ns min
   for (i = 0; i < 250; i++) {
   asm("    nop");
   }
```

```
// Initialization Step 5: Set All Bank Precharge
write_ddramc(ddram_controller_address, HMPDDRC_MR, AT91C_DDRC2_MODE_PRCGALL_CMD);
*((unsigned volatile int*) ddram_address) = 0;
// wait 400 ns min
for (i = 0; i < 250; i++) {
asm("    nop");
}

// Initialization Step 6: Set EMR operation (EMRS2)
write_ddramc(ddram_controller_address, HMPDDRC_MR, AT91C_DDRC2_MODE_EXT_LMR_CMD);
*((unsigned int *)(ddram_address + 0x4000000)) = 0;
// wait 2 cycles min
for (i = 0; i < 100; i++) {
asm("    nop");
}

// Initialization Step 7: Set EMR operation (EMRS3)
write_ddramc(ddram_controller_address, HMPDDRC_MR, AT91C_DDRC2_MODE_EXT_LMR_CMD);
*((unsigned int *)(ddram_address + 0x6000000)) = 0;
// wait 2 cycles min
for (i = 0; i < 100; i++) {
asm("    nop");
}

// Initialization Step 8: Set EMR operation (EMRS1)
write_ddramc(ddram_controller_address, HMPDDRC_MR, AT91C_DDRC2_MODE_EXT_LMR_CMD);
*((unsigned int *)(ddram_address + 0x2000000)) = 0;
// wait 200 cycles min
for (i = 0; i < 10000; i++) {
asm("    nop");
}

// Initialization Step 9: enable DLL reset
cr = read_ddramc(ddram_controller_address, HMPDDRC_CR);
write_ddramc(ddram_controller_address, HMPDDRC_CR, cr | AT91C_DDRC2_DLL_RESET_ENABLED);

// Initialization Step 10: reset DLL
write_ddramc(ddram_controller_address, HMPDDRC_MR, AT91C_DDRC2_MODE_EXT_LMR_CMD);
*(((unsigned volatile int*) ddram_address)) = 0;
// wait 2 cycles min
for (i = 0; i < 100; i++) {
asm("    nop");
}
```

```
// Initialization Step 11: Set All Bank Precharge
write_ddramc(ddram_controller_address, HMPDDRC_MR, AT91C_DDRC2_MODE_PRCGALL_CMD);
*(((unsigned volatile int*) ddram_address)) = 0;
// wait 400 ns min
for (i = 0; i < 250; i++) {
asm("    nop");
}

// Initialization Step 12: Two auto-refresh (CBR) cycles are provided. Program the auto refresh
// command (CBR) into the Mode Register.
write_ddramc(ddram_controller_address, HMPDDRC_MR, AT91C_DDRC2_MODE_RFSH_CMD);
*(((unsigned volatile int*) ddram_address)) = 0;
// wait 10 cycles min
for (i = 0; i < 100; i++) {
asm("    nop");
}
// Set 2nd CBR
write_ddramc(ddram_controller_address, HMPDDRC_MR, AT91C_DDRC2_MODE_RFSH_CMD);
*(((unsigned volatile int*) ddram_address)) = 0;
// wait 10 cycles min
for (i = 0; i < 100; i++) {
asm("    nop");
}

// Initialization Step 13: Program DLL field into the Configuration Register to low (Disable
// DLL reset).
cr = read_ddramc(ddram_controller_address, HMPDDRC_CR);
write_ddramc(ddram_controller_address, HMPDDRC_CR, cr & (~AT91C_DDRC2_DLL_RESET_ENABLED));

// Initialization Step 14: A Mode Register set (MRS) cycle is issued to program the parameters
// of the DDR2-SDRAM devices.
write_ddramc(ddram_controller_address, HMPDDRC_MR, AT91C_DDRC2_MODE_LMR_CMD);
*(((unsigned volatile int*) ddram_address)) = 0;

// Initialization Step 15: Program OCD field into the Configuration Register to high (OCD
// calibration default).
cr = read_ddramc(ddram_controller_address, HMPDDRC_CR);
write_ddramc(ddram_controller_address, HMPDDRC_CR, cr | AT91C_DDRC2_OCD_DEFAULT);

// Initialization Step 16: An Extended Mode Register set (EMRS1) cycle is issued to OCD default
// value.
write_ddramc(ddram_controller_address, HMPDDRC_MR, AT91C_DDRC2_MODE_EXT_LMR_CMD);
*(((unsigned int*) (ddram_address + 0x2000000))) = 0;
// wait 2 cycles min
for (i = 0; i < 100; i++) {
asm("    nop");
}
```

```
    // Initialization Step 17: Program OCD field into the Configuration Register to low (OCD
    // calibration mode exit). Write a 1 to DIC_DS field to use DDR2 weak drive strength.
    cr = read_ddramc(ddram_controller_address, HMPDDRC_CR);
    write_ddramc(ddram_controller_address, HMPDDRC_CR, cr & (~AT91C_DDRC2_OCD_EXIT));
    write_ddramc(ddram_controller_address, HMPDDRC_CR, cr |(AT91C_DDRC2_WEAKSTRENGTH));

    // Initialization Step 18: An Extended Mode Register set (EMRS1) cycle is issued to enable OCD
    // exit.
    write_ddramc(ddram_controller_address, HMPDDRC_MR, AT91C_DDRC2_MODE_EXT_LMR_CMD);
    *(((unsigned int*) (ddram_address + 0x6000000))) = 0;
    // wait 2 cycles min
    for (i = 0; i < 100; i++) {
    asm("    nop");
    }

    // Initialization Step 19, 20: A mode Normal command is provided. Program the Normal mode into
    // Mode Register.
    write_ddramc(ddram_controller_address, HMPDDRC_MR, AT91C_DDRC2_MODE_NORMAL_CMD);
    *(((unsigned volatile int*) ddram_address)) = 0;

    // Initialization Step 21: Write the refresh rate into the count field in the Refresh Timer
    // Register.
    // Set Refresh timer
    write_ddramc(ddram_controller_address, HMPDDRC_RTR, ddram_config->ddramc_rtr);
    // OK, now we are ready to work on the DDRSDR
    // wait for the end of calibration
    for (i = 0; i < 500; i++) {
    asm("    nop");
    }
    return 0;
}
```

Appendix C. LPDDR2-SDRAM Initialization Code Example

This appendix provides an example of the LPDDR2 initialization code, associated with the different steps of the LPDDR2-SDRAM initialization sequence.

```
void LPDDR2_MT42L128M16D1_Initialise( LPDDR2 psst_ddr2 )
{

/*******************************************************************************************/
/*******************************************************************************************/
// Initialization Step 1
// Program the memory device type into the Memory Device Register
/*******************************************************************************************/
/*******************************************************************************************/
  // Memory device = LPDDR2 => MPDDRC_MD_MD_LPDDR2_SDRAM
  // Data bus width = 32 bits => 0x0 (The system is in 64 bits, thus memory data bus width should
be 32 bits)
  MPDDRC->MPDDRC_MD = MPDDRC_MD_MD_LPDDR2_SDRAM ;// LPDDR2

/*******************************************************************************************/
/*******************************************************************************************/
// Initialization Step 2
// Program the features of Low-power DDR2-SDRAM device into the Timing Register
// (asynchronous timing, trc, tras, etc.) and into the Configuration Register (number of
// columns, rows, banks, CAS latency and output drive strength) (see Section 8.3 on
// page 35, Section 8.4 on page 39 and Section 80.5 on page 41).
/*******************************************************************************************/
/*******************************************************************************************/
  ////////////////////////MPDDRC Configuration Register//////////////////////////
    // NC = 0x0. Number of collumn to address is 9 (extract from memory data sheet)
    // NR = 0x2. Number of row to address is 13 (extract from memory data sheet)
    // CAS latency = 3. FPGA platform runs at 30 MHz (depends on the frequency, check memory data
sheet)
    // No DLL in LPDDR2 devices => DLL, DIS_DLL and DIC_DS = 0
    // ZQ = 0. ZQ_INIT calibration will be performed later
    // OCD = 0
    // DQMS = 0. Bus isnt shared
    // NB = 0.5 8 banks (extract from memory data sheet)
    // NDQS = 0. LPDDR2 uses DQS and NDQS
    // DECOD = 0. Sequential decoding is choosen (may changed after the initialization)
    // UNAL = 1; Unaligned accesses will be performed
  MPDDRC->MPDDRC_CR = (psst_ddr2.n_col|     // 9 col + 8 COl supported or not
          psst_ddr2.n_row  |                // 14 row
              MPDDRC_CR_CAS_3_LPDDR2    |   // CAS 3
              MPDDRC_CR_NB_8) |             // 8 banks
              MPDDRC_CR_UNAL_SUPPORTED|     // Unaligned accesses
              MPDDRC_CR_ENRDM_ON;
```

```
// Write the LPDDR2 drive strength according to the PCB. It should be set according to RDIV field
in MPDDRC I/O Calibration Register
// DS Write-only OP<3:0>
      // 0000B: reserved
      // 0001B: 34.3-ohm typical
      // 0010B: 40-ohm typical (default)
      // 0011B: 48-ohm typical
      // 0100B: 60-ohm typical
      // 0101B: reserved for 68.6-ohm typical
      // 0110B: 80-ohm typical
      // 0111B: 120-ohm typical (optional)
      // All others: reserved

    MPDDRC->MPDDRC_LPDDR2_LPR|=MPDDRC_LPDDR2_LPR_DS(0x3);


  //////////////////////////MPDDRC Timing Parameter   //////////////////////////

MPDDRC->MPDDRC_TPR0  = \
MPDDRC_TPR0_TRAS(psst_ddr2.t_tras)  | /*03-TRAS tRAS Row active time*/  \
MPDDRC_TPR0_TRCD(psst_ddr2.t_trcd)  | /*04 -TRC tRCD RAS-to-CAS delay*/  \
MPDDRC_TPR0_TWR(psst_ddr2.t_twr)    | /*05 -TWR tWR WRITE recovery time */   \
MPDDRC_TPR0_TRC(psst_ddr2.t_trc)  |/*06 -TRC tRC ACTI-to-ACTIVT command period*/  \
MPDDRC_TPR0_TRP(psst_ddr2.t_trp) |/*07 -TRP tRPpb  Row precharge time  */  \
MPDDRC_TPR0_TRRD(psst_ddr2.t_trrd) |/*08 -TRRD tRRD Active bank a to active bank b*/  \
MPDDRC_TPR0_TWTR(psst_ddr2.t_twtr) | /*09 -TWTR-tWTR Internal WRITE-to-READcommand delay*/  \
MPDDRC_TPR0_TMRD(psst_ddr2.t_tmrd)/*10 -TMRD-tMRD  */;

MPDDRC->MPDDRC_TPR1  =  \
MPDDRC_TPR1_TRFC(psst_ddr2.t_trfc)  |/*11 -TRFC tRFCab Refresh cycle time  */ \
MPDDRC_TPR1_TXSNR(psst_ddr2.t_txsnr) | /*12 -TXSNR   SELF REFRESH exit to next valid delay */\
MPDDRC_TPR1_TXSRD(psst_ddr2.t_txsrd)  | /*13-TXSRD Exit Self Refresh*/\
MPDDRC_TPR1_TXP(psst_ddr2.t_txp) /*14 -TXP-tXP Exit power-down */     ;

MPDDRC->MPDDRC_TPR2  =  \
MPDDRC_TPR2_TXARD(psst_ddr2.t_txard) |/*15 TXARD-txARD   */ \
MPDDRC_TPR2_TXARDS(psst_ddr2.t_tards) |/*16 TXARDS-txARDs */ \
MPDDRC_TPR2_TRPA(psst_ddr2.t_trpa)   |/*17 TRPA-tRPpab Row precharge time (all banks)  */  \
MPDDRC_TPR2_TRTP(psst_ddr2.t_trtp) |/*18 TRTP-tRTP */  \
MPDDRC_TPR2_TFAW(psst_ddr2.t_tfaw) /*19TFAW--tFAW */;

MPDDRC->MPDDRC_LPR= 0x00000000; // Set low power register to normal mode

/*****************************************************************************/
/*****************************************************************************
// Initialization Step 3
// An NOP command is issued to the Low-power DDR2-SDRAM. Program the NOP
```

```
// command into the Mode Register, the application must set the MODE (MDDRC Command
// Mode) field to 1 in the Mode Register (see Section 8.1 on page 32). Perform a
// write access to any Low-power DDR2-SDRAM address to acknowledge this command.
// Now, clocks which drive Low-power DDR2-SDRAM devices are enabled.
// A minimum pause of 100 ns must be observed to precede any signal toggle.
*********************************************************************************
*********************************************************************************/
   MPDDRC->MPDDRC_MR= MPDDRC_MR_MODE_NOP_CMD;// NOP to ENABLE CLOCK output
   *(unsigned int *)DDR_CS_ADDR= 0x00000000;// Access to memory
   Wait (0xFFFF);// Delay loop (at least 100 ns)

/*********************************************************************************
*********************************************************************************
// Initialization Step 4
// An NOP command is issued to the Low-power DDR2-SDRAM. Program the NOP
// command into the Mode Register, the application must set MODE to 1 in the Mode
// Register (see Section 8.1 on page 32). Perform a write access to any Low-power
// DDR2-SDRAM address to acknowledge this command. Now, CKE is driven high.
// A minimum pause of 200 ìs must be satisfied before Reset Command.
*********************************************************************************
*********************************************************************************/
     MPDDRC->MPDDRC_MR= MPDDRC_MR_MODE_NOP_CMD;// NOP to drive CKE high
   *(unsigned int *)DDR_CS_ADDR= 0x00000000;// Access to memory
       Wait (0xFFFF);// Delay loop (at least 200 us)

/*********************************************************************************
*********************************************************************************
// Initialization Step 5
// A reset command is issued to the Low-power DDR2-SDRAM. Program
// LPDDR2_CMD in the MODE (MDDRC Command Mode) and MRS (Mode Register
// Select LPDDR2) field of the Mode Register, the application must set MODE to 7 and
// MRS to 63. (see Section 8.1 on page 32). Perform a write access to any Low-power
// DDR2-SDRAM address to acknowledge this command. Now, the reset command is issued.
// A minimum pause of 1 ìs must be satisfied before any commands.
*********************************************************************************
*********************************************************************************/
   MPDDRC->MPDDRC_MR=MPDDRC_MR_MRS( 0x3F)| MPDDRC_MR_MODE_LPDDR2_CMD;/// Reset command. MODE =
0x7 and MRS = 0x3F
   *(unsigned int *)DDR_CS_ADDR= 0x00000000;// Access to memory
     Wait (0xFFFF);// Delay loop (at least 1 us)

/*********************************************************************************
*********************************************************************************
// Initialization Step 6
// A Mode Register Read command is issued to the Low-power DDR2-SDRAM. Program
// LPPDR2_CMD in the MODE and MRS field of the Mode Register, the
// application must set MODE to 7 and must set MRS field to 0. (see Section 8.1 on
```

```
// page 32). Perform a write access to any Low-power DDR2-SDRAM address to
// acknowledge this command. Now, the Mode Register Read command is issued.
// A minimum pause of 10 is must be satisfied before any commands.
*******************************************************************************
*******************************************************************************/
  // Mode Register Read  command. MODE = 0x7 and MRS = 0x00
    MPDDRC->MPDDRC_MR= MPDDRC_MR_MODE_LPDDR2_CMD |MPDDRC_MR_MRS( 0x00);
    *(unsigned int *)DDR_CS_ADDR= 0x00000000;// Access to memory
    Wait (0xFFFF);// Delay loop (at least 1 us)

/*******************************************************************************
*******************************************************************************
// Initialization Step 7
A calibration command is issued to the Low-power DDR2-SDRAM. Program the type
of calibration into the Configuration Register, ZQ field, RESET value (see Section 8.3
”MPDDRC Configuration Register” on page 37). In the Mode Register, program the
MODE field to LPDDR2_CMD value, and the MRS field; the application must set
MODE to 7 and MRS to 10 (see Section 8.1 ”MPDDRC Mode Register” on page 34).
Perform a write access to any Low-power DDR2-SDRAM address to acknowledge
this command. Now, the ZQ Calibration command is issued. Program the type of calibration
into the Configuration Register, ZQ field,
*******************************************************************************
*******************************************************************************/
   MPDDRC->MPDDRC_CR&=~MPDDRC_CR_ZQ_Msk;
         MPDDRC->MPDDRC_CR|=   MPDDRC_CR_ZQ_RESET;
        // Mode Register Read  command. MODE = 0x7 and MRS = 0x0A

    MPDDRC->MPDDRC_MR=  MPDDRC_MR_MODE_LPDDR2_CMD |MPDDRC_MR_MRS( 0x0A);
    *(unsigned int *)DDR_CS_ADDR= 0x00000000;// Access to memory
    Wait (0xFFFF);// Delay loop (at least 1 us)
    MPDDRC->MPDDRC_CR&=~MPDDRC_CR_ZQ_Msk;
    MPDDRC->MPDDRC_CR|=   MPDDRC_CR_ZQ_SHORT;

/*******************************************************************************
*******************************************************************************
// Initialization Step 8
// A Mode Register Write command is issued to the Low-power DDR2-SDRAM. Program
// LPPDR2_CMD in the MODE and MRS field in the Mode Register, the
// application must set MODE to 7 and must set MRS field to 0.5 (see Section 8.1 on
// page 32). The Mode Register Write command cycle is issued to program the parameters
// of the Low-power DDR2-SDRAM devices, in particular burst length. Perform a
// write access to any Low-power DDR2-SDRAM address to acknowledge this command.
// Now, the Mode Register Write command is issued.
*******************************************************************************
*******************************************************************************/
  // Programm LPDDR2 parameters MODE = 0x7 and MRS = 0x01
    MPDDRC->MPDDRC_MR= MPDDRC_MR_MODE_LPDDR2_CMD |MPDDRC_MR_MRS( 0x01);
```

```
   *(unsigned int *)DDR_CS_ADDR= 0x00000000;// Access to memory
   Wait (0xFFFF);// Add a delay loop (not is the programmer datasheet)

/*******************************************************************************
********************************************************************************
// Initialization Step 9
// Mode Register Write Command is issued to the Low-power DDR2-SDRAM. Program
// LPPDR2_CMD in the MODE and MRS field in the Mode Register, the
// application must set MODE to 7 and must set MRS field to 2. (see Section 8.1 on
// page 32). The Mode Register Write command cycle is issued to program the parameters
// of the Low-power DDR2-SDRAM devices, in particular CAS latency. Perform a
// write access to any Low-power DDR2-SDRAM address to acknowledge this command.
// Now, the Mode Register Write command is issued.
********************************************************************************
********************************************************************************/
  // Programm LPDDR2 CAS MODE = 0x7 and MRS = 0x02
    MPDDRC->MPDDRC_MR= MPDDRC_MR_MODE_LPDDR2_CMD |MPDDRC_MR_MRS( 0x02);
    *(unsigned int *)DDR_CS_ADDR= 0x00000000;// Access to memory
    Wait (0xFFFF);// Add a delay loop (not is the programmer datasheet)

/*******************************************************************************
********************************************************************************
// Initialization Step 10
// A Mode Register Write Command is issued to the Low-power DDR2-SDRAM. Program
// LPPDR2_CMD in the MODE and MRS field of the Mode Register, the
// application must set MODE to 7 and must set MRS field to 3. (see Section 8.1 on
// page 32). The Mode Register Write command cycle is issued to program the parameters
// of the Low-power DDR2-SDRAM devices, in particular Drive Strength and Slew
// Rate. Perform a write access to any Low-power DDR2-SDRAM address to acknowledge
// this command. Now, the Mode Register Write command is issued.
********************************************************************************
********************************************************************************/
  // Programm LPDDR2 DS MODE = 0x7 and MRS = 0x03
    MPDDRC->MPDDRC_MR= MPDDRC_MR_MODE_LPDDR2_CMD |MPDDRC_MR_MRS( 0x03);//0x00000307;
    *(unsigned int *)DDR_CS_ADDR= 0x00000000;// Access to memory
    Wait (0xFFFF);// Add a delay loop (not is the programmer datasheet)

/*******************************************************************************
********************************************************************************
// Initialization Step 11
// A Mode Register Write Command is issued to the Low-power DDR2-SDRAM. Program
// LPPDR2_CMD in the MODE and MRS field of the Mode Register, the
// application must set MODE to 7 and must set MRS field to 16. (see Section 8.1 on
// page 32). Mode Register Write command cycle is issued to program the parameters
// of the Low-power DDR2-SDRAM devices, in particular Partial Array Self Refresh
// (PASR). Perform a write access to any Low-power DDR2-SDRAM address to
// acknowledge this command. Now, the Mode Register Write command is issued.
```

```
*******************************************************************************
*******************************************************************************/
  // Programm LPDDR2 PASR MODE = 0x7 and MRS = 0x10
    MPDDRC->MPDDRC_MR=MPDDRC_MR_MODE_LPDDR2_CMD |MPDDRC_MR_MRS( 0x10);// 0x00001007;
    *(unsigned int *)DDR_CS_ADDR= 0x00000000;// Access to memory
    Wait (0xFFFF);// Add a delay loop (not is the programmer datasheet)

/*******************************************************************************
*******************************************************************************
// Initialization Step 12
// Write the refresh rate into the COUNT field in the Refresh Timer register (see page
// 33). (Refresh rate = delay between refresh cycles). The Low-power DDR2-SDRAM
// device requires a refresh every 7.81 ìs. With a 100 MHz frequency, the refresh timer
// count register must to be set with (7.81/100 MHz) = 781 i.e. 0x030d.
*******************************************************************************
*******************************************************************************/

         MPDDRC->MPDDRC_RTR&=~MPDDRC_RTR_COUNT_Msk;
         MPDDRC->MPDDRC_RTR |=MPDDRC_RTR_COUNT(psst_ddr2.t_refresh);
         //MPDDRC->MPDDRC_RTR|= MPDDRC_RTR_ADJ_REF ;// MR4 READ enabled

    MPDDRC->MPDDRC_MR= 0x00000000;// Set Normal mode
    *(unsigned int *)DDR_CS_ADDR= 0x00000000;// Perform

     Wait (0xFFFF);
    // Launch short ZQ calibration

    MPDDRC->MPDDRC_CR&= ~   (MPDDRC_CR_ZQ_Msk);// Enable short calibration in the CR
         MPDDRC->MPDDRC_CR |= (MPDDRC_CR_ZQ_SHORT);
         MPDDRC->MPDDRC_CR |= MPDDRC_CR_DLL_RESET_ENABLED;

       *(unsigned int *)DDR_CS_ADDR= 0x00000000;// Perform
    // Calculate ZQS: search for tZQCS in the memory datasheet => tZQCS = 180 ns
    MPDDRC->MPDDRC_LPDDR2_TIM_CAL =  MPDDRC_LPDDR2_TIM_CAL_ZQCS(psst_ddr2.t_tZQCS);

}
```

Revision History

Table 7-2. Implementation of SDRAM on SAMA5D2x Devices Revision History

Doc. Rev.	Date	Changes
44044A	07-Sep-15	First issue

Atmel Corporation 1600 Technology Drive, San Jose, CA 95110 USA **T:** (+1)(408) 441.0311 **F:** (+1)(408) 436.4200 | **www.atmel.com**