EXHIBIT 5
(PART 2)

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

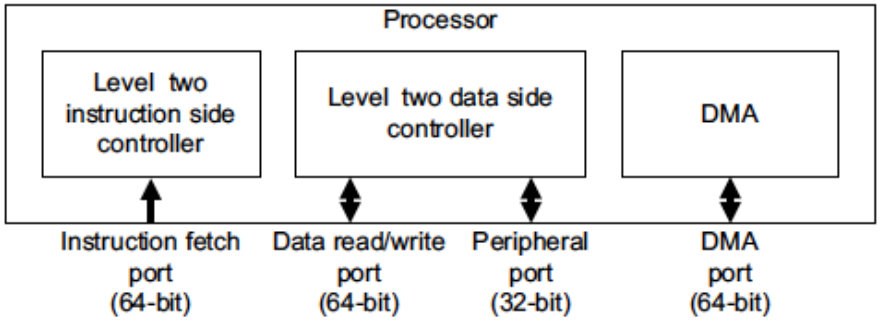


Figure 8-1 Level two interconnect interfaces

ARM11 TRM at 8-2 (389) [TPLBN035476].

1.5.3 Prefetch unit

The prefetch unit fetches instructions from the instruction cache, Instruction TCM, or from external memory and predicts the outcome of branches in the instruction stream.

ARM11 TRM at 1-11 (38) [TPLBN035125].

Instruction fetch

Servicing instruction cache misses and noncacheable instruction fetches.

ARM11 TRM at 1-16 (42) [TPLBN035129].

ARMv6-M is a subset of ARMv7-M, that provides:

- a lightweight version of the ARMv7-M programming model
- the Debug Extension that includes architecture extensions for debug support, see Chapter C1 *ARMv6-M Debug*.
- ARMv6 Thumb 16-bit instruction set compatibility at the application level

ARMv6 Reference Manual at A1-26 (26) [TPLBN035872].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

A5.1 Thumb instruction set encoding

The Thumb instruction stream is a sequence of halfword-aligned halfwords. Each Thumb instruction is either a single 16-bit halfword in that stream, or a 32-bit instruction consisting of two consecutive halfwords in that stream.

ARMv6 Reference Manual at A5-82 (82) [TPLBN035928].

A3.8 Caches and memory hierarchy

Support for caches in ARMv6-M is limited to memory attributes. These can be exported on a supporting bus protocol such as AMBA AHB or AMBA AXI to support system caches.

In situations where a breakdown in coherency can occur, software must manage the caches using cache maintenance operations that are memory mapped and IMPLEMENTATION DEFINED.

A3.8.1 Introduction to caches

A cache is a block of high-speed memory locations containing both address information and the associated data. The purpose is to increase the average speed of a memory access. Caches operate on two principles of locality:

Spatial locality an access to one location is likely to be followed by accesses from adjacent locations, for example sequential instruction execution or usage of a data structure

Temporal locality an access to an area of memory is likely to be repeated within a short time period, for example execution of a code loop.

ARMv6 Reference Manual at A3-63 (63) [TPLBN035909], *see also* A6-137 (137) [TPLBN035983].

Execution stream

The stream of instructions that would have been executed by sequential execution of the program.

ARMv6 Reference Manual at Glossary-428 (428) [(428) [TPLBN036274].

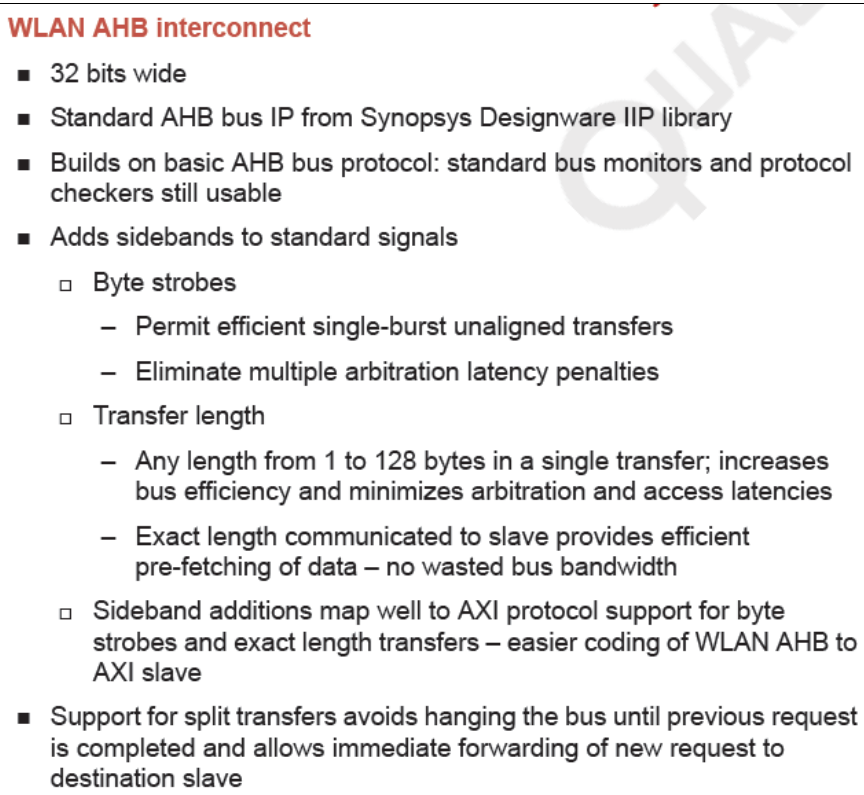
Note

ARMv7-M has limited support for 64-bit integers. Most 64-bit operations require sequences of two or more instructions to synthesize them.

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

	ARMv6 Reference Manual at A2-31 (31) [TPLBN035877]. The Fetch stages can hold up to four instructions, where branch prediction is performed on instructions ahead of execution of earlier instructions. The Issue and Decode stages can contain any instruction in parallel with a predicted branch. The Execute, Memory, and Write stages can contain a predicted branch, an ALU or multiply instruction, a load/store multiple instruction, and a coprocessor instruction in parallel execution. ARM11 TRM at 1-23 (53) [TPLBN035140].
[1f] said bus having a width at least equal to a number of bits in each of the instructions times a number of the instructions fetched in parallel,	On information and belief, in each Accused Product, said bus has a width at least equal to a number of bits in each of the instructions times a number of the instructions fetched in parallel. For example: **1. Qualcomm Accused Products:** As to Accused Products with Qualcomm processors, *see, e.g.*: ▪ Improvements compared to MSM7225 and MSM7200A devices • Bus/processor speed enhancements ◆ 200 MHz AXI and AHB bus ◆ 400 MHz ARM9™ ◆ 600 MHz ARM11™ • 256 kB ARM11 L2 cache • ARM11 floating point Dual Krait µP CPUs, each with: • Up to 1.5 GHz • 1 MB L2 cache • 32 kB L1 instruction and data caches • ARM v7 compliant • TrustZone support • VeNum 128-bit SIMD MM coprocessor

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

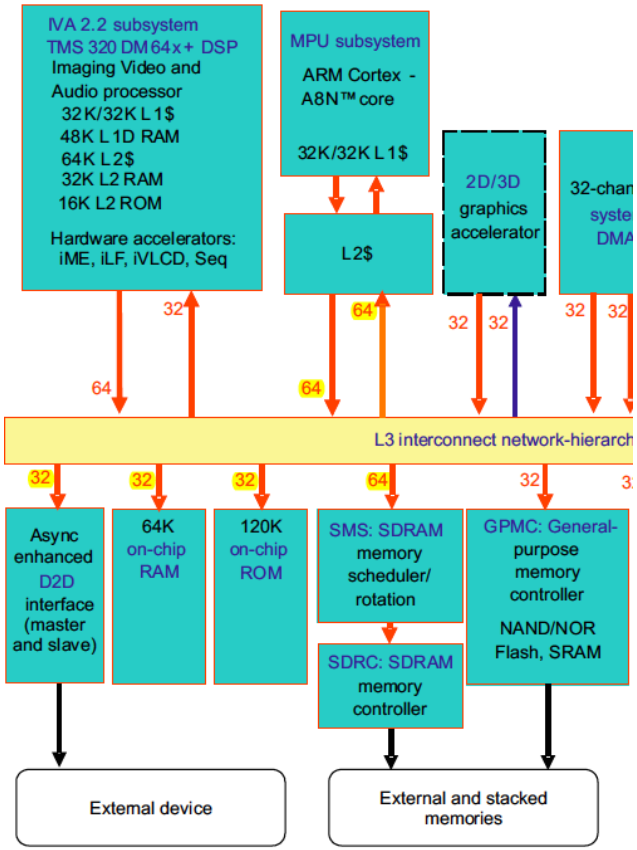


MSM7227 Chipset Training at 5; *see also* MSM8960 Design Guidelines at 454, 86, 88; http://www.anandtech.com/show/4940/qualcomm-new-snapdragon-s4-msm8960-krait-architecture (showing fetch and decode stages for the Krait core; "The architecture can fetch and decode three instructions per clock. The decoders are equally capable of decoding any ARMv7-A instructions."); and the ARM core documentation cited below.

As to Accused Products with OMAP processors and other ARM processors, *see, e.g.*:

2. OMAP Accused Products:

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung



OMAP36xx TRM at 191 [TPLBN004904].

1.4.1 AMBA AXI interface

The AXI bus interface is the main interface to the system bus. It performs L2 cache fills and noncacheable accesses for both instructions and data. The AXI interface supports 64-bit or 128-bit wide input and output data buses. It also supports multiple outstanding requests on the AXI bus. The AXI signals are synchronous to the **CLK** input. A wide range of bus clock to core clock ratios is possible through the use of the AXI clock enable signal **ACLKEN**. See the *AMBA AXI Protocol Specification* for more information.

Cortex-A8 TRM, at 1-7 (32) [TPLBN034309], *see also* Glossary-2 (564) [TPLBN034841].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

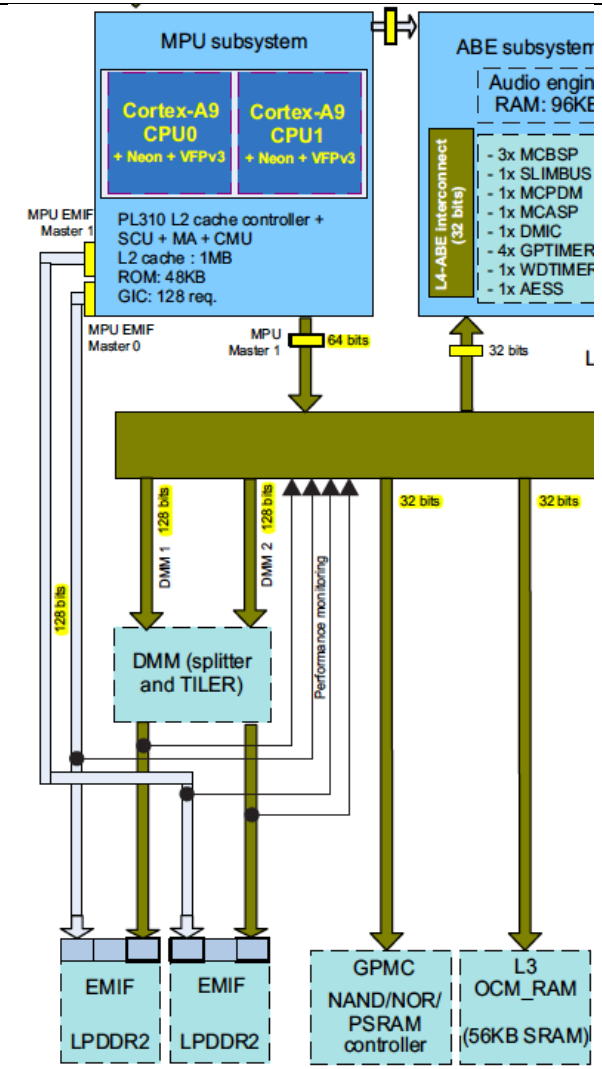
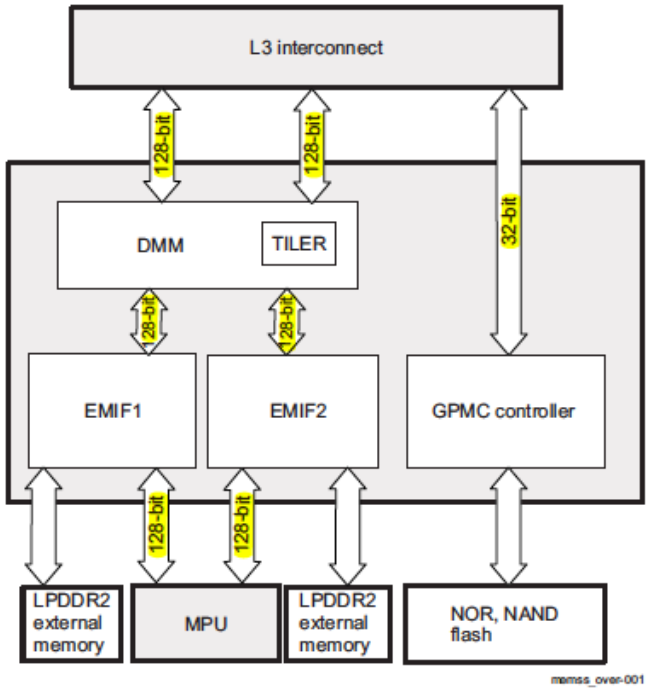


Figure 16-1. Memory Subsystem Functional Diagram



OMAP4470 TRM at 276 [TPLBN021476], 3280 [TPLBN024480].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

	8.1.1 About the Cortex-A9 L2 interface The Cortex-A9 L2 interface consists of two 64-bit wide AXI bus masters: • M0 is the data side bus • M1 is the instruction side bus and has no write channels. Cortex-A9 TRM at 8-2 (128) [TPLBN034985]. The EMIF has the following capabilities: • Supports JEDEC standard-compliant LPDDR2-SDRAM (S2 and S4) devices • 2-GB SDRAM address range over two chip selects (CSs) (configurable with the DMM; see Section 16.2, *Dynamic Memory Manager*, for more information) • Supports two independent CSs, with their corresponding register sets, and independent page tracking • Both CSs must have the same memory type and size. • Flexible address muxing scheme that permits choosing different bank-mapping allocation by configuring the bank, column, and row-address decoding ordering • 16- or 32-bit data path to external SDRAM • Supports LPDDR2 devices with 1, 2, 4, or 8 internal banks • Supports the following data bus widths: – 128-bit level 3 (L3) interconnect data bus width – 16- and 32-bit SDRAM data bus width OMAP4470 TRM at 3282 [TPLBN024482]. **16.1.4.1 GPMC Features** The GPMC is the external memory controller of the device. The GPMC data access engine provides a flexible programming model for communication with all standard memories. The GPMC supports various accesses: • Asynchronous read/write access • Asynchronous read page access (4, 8, and 16 Word16) • Synchronous read/write access • Synchronous read/write burst access without wrap capability (4, 8, and 16 Word16) OMAP4470 TRM at 3283 [TPLBN024483].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

16.1.6 OCM Overview

The on-chip memory (OCM) subsystem consists of the following OCM controllers: one connected to an on-chip ROM (SAR ROM), one connected to an on-chip RAM (SAR RAM), and one connected to an on-chip SRAM (L3 SRAM). Each memory controller has its own dedicated interface to the L3 interconnect.

OMAP4470 TRM at 3284 [TPLBN024484].

16.1.6.1 SAR ROM

This on-chip ROM contains 4KB of memory and a linked list of descriptors used by sDMA during the restore context operation (when the device transitions from off to on mode).

The device-embedded SAR ROM has the following characteristics:

- 4-KB ROM
- 32-bit access per cycle
- Support for single- and burst-access transactions

16.1.6.2 SAR RAM

The on-chip SAR RAM contains 8KB and is mapped as four blocks with irregular region sizes. This memory content is preserved when the device goes into off mode (as long as the wake-up voltage domain remains supplied). It is used as context-saving memory to be written by software so that sDMA restores its saved content when the device transitions from off to on mode.

The device-embedded SAR RAM has the following characteristics:

- Support for single-access transactions
  - Operates at full L4-PER interconnect clock frequency
  - 32-bit access per cycle

16.1.6.3 L3 OCM_RAM

The on-chip L3 OCM_RAM contains 56KB of RAM, and partitioning is defined by the L3 firewall logic. The device-embedded L3 OCM_RAM has the following characteristics:

- Support for single and burst access transactions:
  - Operates at full L3 interconnect clock frequency
  - Fully pipelined, one 32-bit access per cycle
- Restricted access support

OMAP4470 TRM at 3284 [TPLBN024484].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

16.4.4.7 L3 Interconnect Interface

The GPMC L3 interconnect interface is a pipelined interface including an 16 x 32-bit word write buffer.

Any system host can issue external access requests through the GPMC.

The device system can issue the following requests through this interface:

- One 8-bit / 16-bit / 32-bit interconnect access (read/write)
- Two incrementing 32-bit interconnect accesses (read/write)
- Two wrapped 32-bit interconnect accesses (read/write)
- Four incrementing 32-bit interconnect accesses (read/write)
- Four wrapped 32-bit interconnect accesses (read/write)
- Eight incrementing 32-bit interconnect accesses (read/write)
- Eight wrapped 32-bit interconnect accesses (read/write)

OMAP4470 TRM at 3447 [TPLBN024647].

1.1 About the Cortex-A9 processor

The Cortex-A9 processor is a high-performance, low-power, ARM macrocell with an L1 cache subsystem that provides full virtual memory capabilities. The Cortex-A9 processor implements the ARMv7-A architecture and runs 32-bit ARM instructions, 16-bit and 32-bit Thumb instructions, and 8-bit Java™ bytecodes in Jazelle state.

Figure 1-1 shows a Cortex-A9 uniprocessor in a design with a PL390 Interrupt Controller and an L2C-310 L2 Cache Controller,

Cortex-A9 TRM at 1-2 (19) [TPLBN034876], *see also* 1-6 – 1-7 (23-24) [TPLBN034880-81], 3-2 (48) [TPLBN034905], A5-2 (108) [TPLBN034965]; *see also* Cortex-A8 TRM at 2-3 (43) [TPLBN034320].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

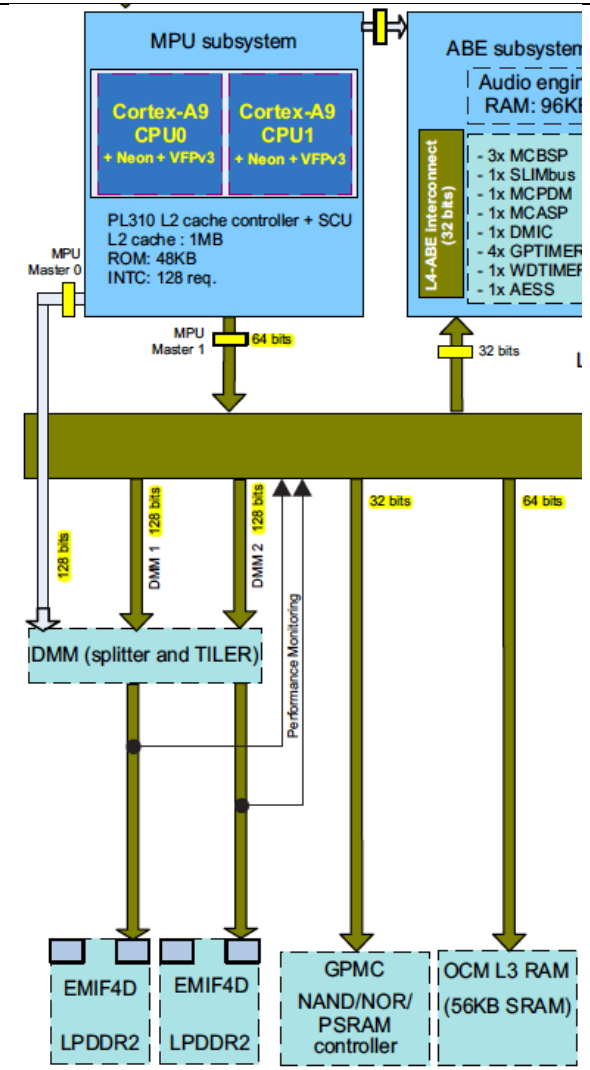
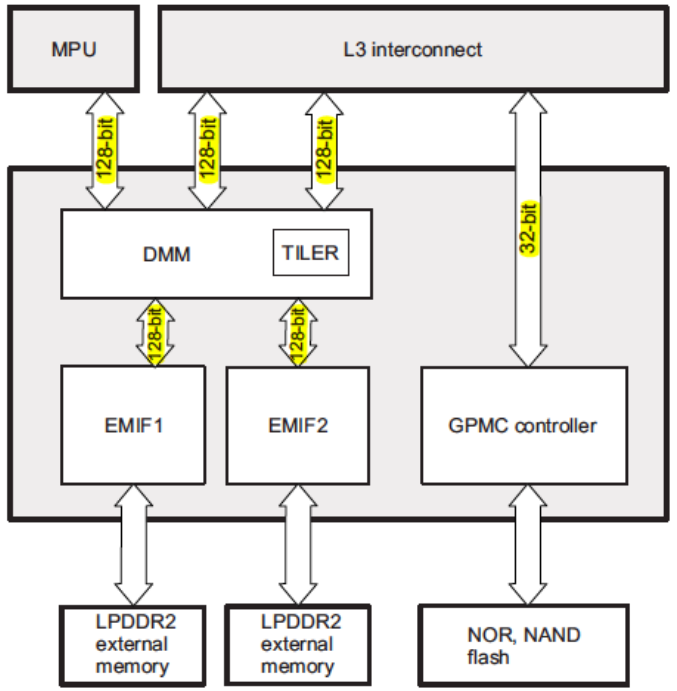


Figure 15-1. Memory Subsystem Functional Diagram



OMAP4430 TRM at 275 [TPLBN008774], 3207 [TPLBN011706].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

1.1 About the Cortex-A9 processor

The Cortex-A9 processor is a high-performance, low-power, ARM macrocell with an L1 cache subsystem that provides full virtual memory capabilities. The Cortex-A9 processor implements the ARMv7-A architecture and runs 32-bit ARM instructions, 16-bit and 32-bit Thumb instructions, and 8-bit Java™ bytecodes in Jazelle state.

Figure 1-1 shows a Cortex-A9 uniprocessor in a design with a PL390 Interrupt Controller and an L2C-310 L2 Cache Controller,

Cortex-A9 TRM at 1-2 (19) [TPLBN034876], *see also* 1-6 – 1-7 (23-24) [TPLBN034880-81], 3-2 (48) [TPLBN034905], A5-2 (108) [TPLBN034965]; *see also* Cortex-A8 TRM at 2-3 (43) [TPLBN034320].

8.1.1 About the Cortex-A9 L2 interface

The Cortex-A9 L2 interface consists of two 64-bit wide AXI bus masters:

- M0 is the data side bus
- M1 is the instruction side bus and has no write channels.

Cortex-A9 TRM at 8-2 (128) [TPLBN034985].

The EMIF has the following capabilities:

- Supports JEDEC standard-compliant LPDDR2-SDRAM (S2 and S4) and LPDDR2-NVM devices
- 2-GB SDRAM address range over two chip selects (CSs) (configurable with the DMM; see Section 15.2, *Dynamic Memory Manager*, for more information)
- Supports two independent CSs, with their corresponding register sets, and independent page tracking
- Both CSs must have the same memory type and size if they are both SDRAM or both nonvolatile memory (NVM). LPDDR2-SDRAM can be used in parallel with LPDDR2-NVM and can have a different size.
- Flexible address muxing scheme which permits choosing different bank-mapping allocation by configuring the bank, column, and row-address decoding ordering
- 16- or 32-bit data path to external SDRAM
- Supports LPDDR2 devices with 1, 2, 4, or 8 internal banks
- Supports the following data bus widths:
  - 128-bit level 3 (L3) interconnect data bus width
  - 16- and 32-bit SDRAM data bus width

OMAP4430 TRM at 3209 [TPLBN011708].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

15.1.4.1 GPMC Features

The GPMC is the external memory controller of the device. The GPMC data access engine provides a flexible programming model for communication with all standard memories. The GPMC supports various accesses:

- Asynchronous read/write access
- Asynchronous read page access (4, 8, and 16 Word16)
- Synchronous read/write access
- Synchronous read/write burst access without wrap capability (4, 8, and 16 Word16)
- Synchronous read/write burst access with wrap capability (4, 8, and 16 Word16)

OMAP4430 TRM at 3210-11 [TPLBN011709-10].

15.1.6 OCM Overview

The on-chip memory (OCM) subsystem consists of the following OCM controllers: one connected to an on-chip ROM (SAR ROM), one connected to an on-chip RAM (SAR RAM), and one connected to an on-chip SRAM (L3 SRAM). Each memory controller has its own dedicated interface to the L3 interconnect.

OMAP4430 TRM at 3211 [TPLBN011710].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

15.1.6.1 SAR ROM

This on-chip ROM contains 4KB of memory and a linked list of descriptors used by sDMA during the restore context operation (when the device transitions from off to on mode).

The device-embedded SAR ROM has the following characteristics:

- 4-KB ROM
- 32-bit access per cycle
- Support for single- and burst-access transactions

15.1.6.2 SAR RAM

The on-chip SAR RAM contains 8K bytes and is mapped as 4 blocks with irregular region sizes. This memory content is preserved when the device goes into off mode (as long as the wake-up voltage domain remains supplied). It is used as context-saving memory to be written by software so that sDMA restores its saved content when the device transitions from off to on mode.

The device-embedded SAR RAM has the following characteristics:

- Support for single-access transactions
  - Operates at full L4-PER interconnect clock frequency
  - 32-bit access per cycle

15.1.6.3 L3 OCM_RAM

The on-chip L3 OCM_RAM contains 56KB of RAM, and partitioning is defined by the L3 firewall logic. The device-embedded L3 OCM_RAM has the following characteristics:

- Support for single and burst access transactions:
  - Operates at full L3 interconnect clock frequency
  - Fully pipelined, one 32-bit access per cycle
- Restricted access support

OMAP4430 TRM at 3211 [TPLBN011710].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

15.4.4.7 L3 Interconnect Interface

The GPMC L3 interconnect interface is a pipelined interface including an 16 x 32-bit word write buffer.

Any system host can issue external access requests through the GPMC.

The device system can issue the following requests through this interface:

- One 8-bit / 16-bit / 32-bit interconnect access (read/write)
- Two incrementing 32-bit interconnect accesses (read/write)
- Two wrapped 32-bit interconnect accesses (read/write)
- Four incrementing 32-bit interconnect accesses (read/write)
- Four wrapped 32-bit interconnect accesses (read/write)
- Eight incrementing 32-bit interconnect accesses (read/write)
- Eight wrapped 32-bit interconnect accesses (read/write)

OMAP4430 TRM at 3375 [TPLBN011874].

3. Samsung S3C6410 Accused Products:

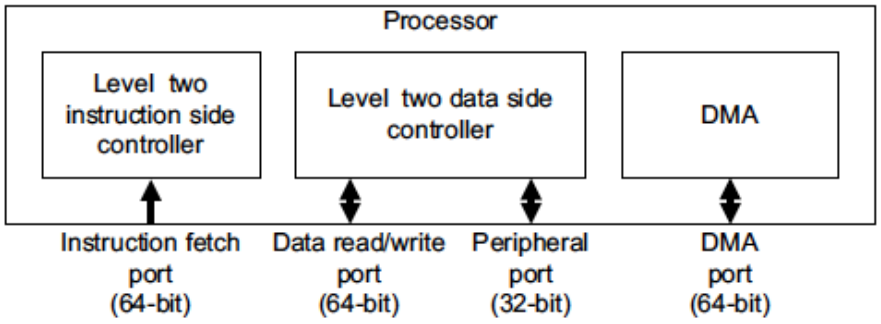


Figure 8-1 Level two interconnect interfaces

ARM11 TRM at 8-2 (389) [TPLBN035476].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

	1.1 ARCHITECTURAL OVERVIEW The S3C6410X is a 16/32-bit RISC microprocessor, which is designed to provide a cost-effective, low-power capabilities, high performance Application Processor solution for mobile phones and general applications. To provide optimized H/W performance for the 2.5G & 3G communication services, the S3C6410X adopts 64/32-bit internal bus architecture. The 64/32-bit internal bus architecture is composed of AXI, AHB and APB buses. It also includes many powerful hardware accelerators for tasks such as motion video processing, audio processing, 2D graphics, display manipulation and scaling. An integrated Multi Format Codec (MFC) supports encoding and decoding of MPEG4/H.263/H.264 and decoding of VC1. This H/W Encoder/Decoder supports real-time video conferencing and TV out for both NTSC and PAL mode. Graphic 3D (hereinafter 3D Engine) is a 3D Graphics Hardware Accelerator which can accelerate OpenGL ES 1.1 & 2.0 rendering. This 3D Engine includes two programmable shaders: one vertex shader and one pixel shader. The S3C6410X has an optimized interface to external memory. This optimized interface to external memory is capable of sustaining the high memory bandwidths required in high-end communication services. The memory system has dual external memory ports, DRAM and Flash/ROM. The DRAM port can be configured to support mobile DDR, DDR, mobile SDRAM and SDRAM. The Flash/ROM port supports NOR-Flash, NAND-Flash, OneNAND, CF and ROM type external memory. S3C6410X User Manual at 1-1 (60) [TPLBN002467].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

1.2.2 MICROPROCESSOR

The ARM1176JZF-S processor incorporates an integer unit that implements the ARM11 ARM architecture v6. It supports the ARM, Thumb™ instruction sets and Jazelle technology to enable direct execution of Java bytecodes, and a range of SIMD DSP instructions that operate on 16-bit or 8-bit data values in 32-bit registers.

The features of ARM1176JZF-S processor include:

- High-speed *Advanced Microprocessor Bus Architecture* (AMBA) *Advanced Extensible Interface* (AXI) level two interfaces supporting prioritized multiprocessor implementations.
- Integer unit with integral EmbeddedICE-RT logic.
- Eight-stage pipeline.
- Branch prediction with return stack.
- Low interrupt latency configuration.
- coprocessors CP14 and CP15.
- Instruction and Data *Memory Management Units* (MMUs), managed using MicroTLB structures backed by a unified Main TLB.
- Instruction and data caches, including a non-blocking data cache with *Hit-Under-Miss* (HUM).
- Virtually indexed and physically addressed caches.
- 64-bit interface to both caches.
- *Vector Floating-Point* (VFP) coprocessor support.

S3C6410X User Manual at 1-4 (63) [TPLBN002470], *see also* ARM11 TRM at 1-2 (28) [TPLBN035115].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

4.2. INTRODUCTION

S3C6410X Memory Sub-system features are as follows:

- Memory Subsystem has one 64-bit AXI slave interface, one 32-bit AXI slave interface, one 32-bit AHB Master interface, two 32-bit AHB slave interfaces, one for data transfer and the other for SFR setting, and one APB interface for DMC SFR setting.
- Memory Subsystem gets booting method and CS selection information from the System Controller.
- Internal AHB data bus connects 32-bit AHB slave data bus with SROMC, two OneNANDC and NFCON.
- Internal AHB SFR bus connects 32-bit AHB slave SFR bus with SROMC, two OneNANDC, CFCON and NFCON.
- Internal AHB master bus is used for CFCON.
- DMC1 uses 64-bit AXI slave interface and APB interface.

S3C6410X User Manual at 4-1 (173) [TPLBN002580].

1.1 About the processor

The ARM1176JZF-S processor incorporates an integer core that implements the ARM11 ARM architecture v6. It supports the ARM and Thumb™ instruction sets, Jazelle technology to enable direct execution of Java bytecodes, and a range of SIMD DSP instructions that operate on 16-bit or 8-bit data values in 32-bit registers.

ARM11 TRM at 1-2 (28) [TPLBN035115], *see also* 2-2 (75) [TPLBN035162].

1.4 ARM1176JZF-S architecture with Jazelle technology

The ARM1176JZF-S processor has three instruction sets:

- the 32-bit ARM instruction set used in ARM state, with media instructions
- the 16-bit Thumb instruction set used in Thumb state
- the 8-bit Java bytecodes used in Jazelle state.

For details of both the ARM and Thumb instruction sets, see the *ARM Architecture Reference Manual*. For full details of the ARM1176JZF-S Java instruction set, see the *Jazelle V1 Architecture Reference Manual*.

ARM11 TRM at 1-6 (32) [TPLBN035119], *see also* 1-10 (36) [TPLBN035123], 2-12 (85) [TPLBN035172].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

1.5.4 Memory system

The level-one memory system provides the core with:

- separate instruction and data caches
- separate instruction and data Tightly-Coupled Memories
- 64-bit datapaths throughout the memory system
- virtually indexed, physically tagged caches
- memory access controls and virtual memory management
- support for four sizes of memory page
- two-channel DMA into TCMs
- I-fetch, D-read/write interface, compatible with multi-layer AMBA AXI
- 32-bit dedicated peripheral interface
- export of memory attributes for second-level memory system.

ARM11 TRM at 1-12 (38) [TPLBN035125], *see also* 7-2 (373) [TPLBN035460].

The processor level two interconnect system uses the following 64-bit wide AXI interfaces:

- Instruction Fetch Interface
- Data Read/Write Interface
- DMA Interface.

Another interface is also provided, the Peripheral Interface. This is a 32-bit AXI interface.

ARM11 TRM at 8-2 (389) [TPLBN035476].

A5.1 Thumb instruction set encoding

The Thumb instruction stream is a sequence of halfword-aligned halfwords. Each Thumb instruction is either a single 16-bit halfword in that stream, or a 32-bit instruction consisting of two consecutive halfwords in that stream.

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

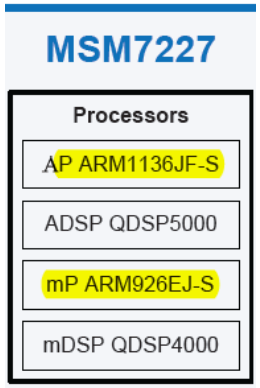
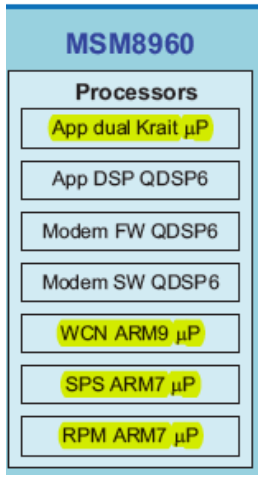
	ARMv6 Reference Manual at A5-82 (82) [TPLBN035928]; *see also* A1-26 (26) [TPLBN035872], A3-63 (63) [TPLBN035909], A6-137 (137) [TPLBN035983].
[1g] said central processing unit integrated circuit including an arithmetic logic unit and a first push down stack connected to said arithmetic logic unit,	On information and belief, each Accused Product has a central processing unit integrated circuit including an arithmetic logic unit and a first push down stack connected to said arithmetic logic unit. For example: **1. Qualcomm Accused products:** As to Accused Products with Qualcomm processors, *see, e.g.*:   MSM7227 Chipset Training at 7; MSM8960 Design Guidelines at 38; http://www.anandtech.com/show/4940/qualcomm-new-snapdragon-s4-msm8960-krait-architecture (Krait similar to ARM core and "capable of decoding any ARMv7-A instructions"). All Qualcomm processors, including those with ARM cores and other cores (*e.g.*, Scorpion, Krait, etc.) contain instruction registers and have pipelines that have multiple stages. *See* http://www.anandtech.com/show/4940/qualcomm-new-snapdragon-s4-msm8960-krait-architecture. *See also* the ARM core documentation cited below. As to Accused Products with ARM processors, *see, e.g.*:

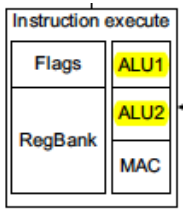
EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

2. OMAP Accused Products:

CALU— Central arithmetic logic unit

CPU— Central processing unit: The CPU is the portion of the processor involved in arithmetic, shifting, and Boolean logic operations, as well as the generation of data and program memory addresses. The CPU includes the CALU, the multiplier, and the ARAU.

OMAP36xx TRM at 3759 [TPLBN008472].



1.3.3 Instruction execute

The instruction execute unit consists of two symmetric *Arithmetic Logical Unit* (ALU) pipelines, an address generator for load and store instructions, and the multiply pipeline. The execute pipelines also perform register write back.

Cortex-A8 TRM at 1-4 – 1-6 (29-31) [TPLBN034306-08].

In addition, the ARM architecture gives you:

- control over both the *Arithmetic Logic Unit* (ALU) and shifter in every data-processing instruction to maximize the use of an ALU and a shifter

ARM Architecture Reference Manual at A1-2 [PIC00005214].

The Arithmetic Logic Unit (ALU) derives its two inputs from the 'Top item' (Rn) and the 'Next item' (Rm) of the 'Push Down Stack' from the 'General Purpose Registers' (the holding place for items placed there by stack operations) and directs its output (Rd) back to the 'General Purpose Registers'.

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

A8.6.2 ADC (register)

Add with Carry (register) adds a register value, the carry flag value, and an optionally-shifted register value, and writes the result to the destination register. It can optionally update the condition flags based on the result.

Encoding T1 ARMv4T, ARMv5T*, ARMv6*, ARMv7

```
ADCS <Rdn>,<Rm>                                   Outside IT block.
ADC<c> <Rdn>,<Rm>                                 Inside IT block.
```

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	0	0	1	0	1	Rm			Rdn		

```
d = UInt(Rdn);  n = UInt(Rdn);  m = UInt(Rm);  setflags = !InITBlock();
(shift_t, shift_n) = (SRType_LSL, 0);
```

Encoding T2 ARMv6T2, ARMv7

```
ADC{S}<c>.W <Rd>,<Rn>,<Rm>{,<shift>}
```

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	0	1	0	1	1	0	1	0	S	Rn				(0)	imm3			Rd				imm2		type		Rm			

```
d = UInt(Rd);  n = UInt(Rn);  m = UInt(Rm);  setflags = (S == '1');
(shift_t, shift_n) = DecodeImmShift(type, imm3:imm2);
if BadReg(d) || BadReg(n) || BadReg(m) then UNPREDICTABLE;
```

Encoding A1 ARMv4*, ARMv5T*, ARMv6*, ARMv7

```
ADC{S}<c> <Rd>,<Rn>,<Rm>{,<shift>}
```

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cond				0	0	0	0	1	0	1	S	Rn				Rd				imm5				type		0	Rm				

```
if Rd == '1111' && S == '1' then SEE SUBS PC, LR and related instructions;
d = UInt(Rd);  n = UInt(Rn);  m = UInt(Rm);  setflags = (S == '1');
(shift_t, shift_n) = DecodeImmShift(type, imm5);
```

ARMv7 Reference Manual at A8-16 (328) [TPLBN049616].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

Assembler syntax

ADC{S}<c><q> {<Rd>,} <Rn>, <Rm> {,<shift>}

where:

S	If S is present, the instruction updates the flags. Otherwise, the flags are not updated.
<c><q>	See *Standard assembler syntax fields* on page A8-7.
<Rd>	The destination register.
<Rn>	The first operand register.
<Rm>	The optionally shifted second operand register.
<shift>	The shift to apply to the value read from <Rm>. If present, encoding T1 is not permitted. If absent, no shift is applied and any encoding is permitted. *Shifts applied to a register* on page A8-10 describes the shifts and how they are encoded.

ARMv7 Reference Manual at A8-17 (329) [TPLBN049617].

The Register File includes register R13, also known as SP or the Stack pointer, which is a pointer to the active stack.

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

A2.3 ARM core registers

In the application level view, an ARM processor has:

- thirteen general-purpose32-bit registers, R0 to R12
- three 32-bit registers, R13 to R15, that sometimes or always have a special use.

Registers R13 to R15 are usually referred to by names that indicate their special uses:

SP, the Stack Pointer

Register R13 is used as a pointer to the active stack.

In Thumb code, most instructions cannot access SP. The only instructions that can access SP are those designed to use SP as a stack pointer.

The use of SP for any purpose other than as a stack pointer is deprecated.

Note

Using SP for any purpose other than as a stack pointer is likely to break the requirements of operating systems, debuggers, and other software systems, causing them to malfunction.

LR, the Link Register

Register R14 is used to store the return address from a subroutine. At other times, LR can be used for other purposes.

When a `BL` or `BLX` instruction performs a subroutine call, LR is set to the subroutine return address. To perform a subroutine return, copy LR back to the program counter. This is typically done in one of two ways, after entering the subroutine with a `BL` or `BLX` instruction:

- Return with a `BX LR` instruction.
- On subroutine entry, store LR to the stack with an instruction of the form:
  `PUSH {<registers>,LR}`
  and use a matching instruction to return:
  `POP {<registers>,PC}`

ThumbEE checks and handler calls use LR in a similar way. For details see Chapter A9 *ThumbEE*.

ARMv7 Reference Manual at A2-11 (43) [TPLBN049331]; *see also* Cortex-A8 TRM at 2-18 (58) [TPLBN034335].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

A6.1.3 Use of 0b1101 as a register specifier

R13 is defined in the Thumb instruction set so that its use is primarily as a stack pointer, and R13 is normally identified as SP in Thumb instructions. In 32-bit Thumb instructions, if you use R13 as a general-purpose register beyond the architecturally defined constraints described in this section, the results are UNPREDICTABLE.

The restrictions applicable to R13 are described in:

- *R13[1:0] definition*
- *32-bit Thumb instruction support for R13.*

See also *16-bit Thumb instruction support for R13* on page A6-5.

ARMv7 Reference Manual at A6-4 (242) [TPLBN049530].

The top two items in the stack are connected to provide inputs into the ALU by using a "POP".

A8.6.122 POP

Pop Multiple Registers loads multiple registers from the stack, loading from consecutive memory locations starting at the address in SP, and updates SP to point just above the loaded data.

ARMv7 Reference Manual at A8-246 (558) [TPLBN049846].

The output of the ALU is connected to the top of the stack and may be saved with a "PUSH."

A8.6.123 PUSH

Push Multiple Registers stores multiple registers to the stack, storing to consecutive memory locations ending just below the address in SP, and updates SP to point to the start of the stored data.

ARMv7 Reference Manual at A8-248 (560) [TPLBN049848].

In the alternative, ARM Cortex A9 processor also includes an operand stack for Jazelle DBX instructions.

A2.10.2 Jazelle direct bytecode execution support

From ARMv5TEJ, the architecture requires every system to include an implementation of the Jazelle extension. The Jazelle extension provides architectural support for hardware acceleration of bytecode execution by a *Java Virtual Machine* (JVM).

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

ARMv7 Reference Manual at A2-73 (105) [TPLBN049393].

Reads

Reads are defined as memory operations that have the semantics of a load.

The memory accesses of the following instructions are reads:

- LDR, LDRB, LDRH, LDRSB, and LDRSH
- LDRT, LDRBT, LDRHT, LDRSBT, and LDRSHT
- LDREX, LDREXB, LDREXD, and LDREXH
- LDM, LDRD, POP, and RFE
- LDC, LDC2, VLDM, VLDR, VLD1, VLD2, VLD3, and VLD4
- the return of status values by STREX, STREXB, STREXD, and STREXH
- in the ARM instruction set only, SWP and SWPB
- in the Thumb instruction set only, TBB and TBH.

Hardware-accelerated opcode execution by the Jazelle extension can cause a number of reads to occur, according to the state of the operand stack and the implementation of the Jazelle hardware acceleration.

Writes

Writes are defined as memory operations that have the semantics of a store.

The memory accesses of the following instructions are Writes:

- STR, STRB, and STRH
- STRT, STRBT, and STRHT
- STREX, STREXB, STREXD, and STREXH
- STM, STRD, PUSH, and SRS
- STC, STC2, VSTM, VSTR, VST1, VST2, VST3, and VST4
- in the ARM instruction set only, SWP and SWPB.

Hardware-accelerated opcode execution by the Jazelle extension can cause a number of writes to occur, according to the state of the operand stack and the implementation of the Jazelle hardware acceleration.

ARMv7 Reference Manual at A3-42 (156) [TPLBN049444].

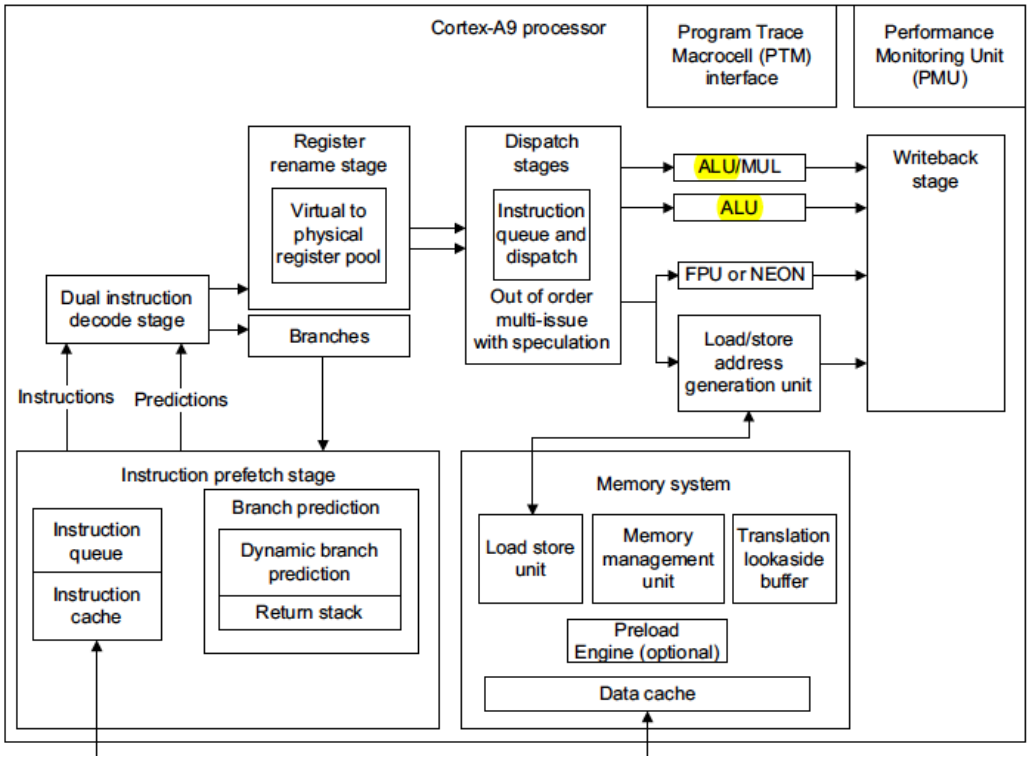
The operand stack contains a top item register and a next item register connected to provide inputs to the ALU

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

	and the output of the ALU is connected to the top item register. The first four elements of the stack are held in the register file in registers R0-R3. The stack pointer is held in register R6. In Java State, the processor assigns several ARM registers to functions specific to the Java machine (e.g., R6= stack pointer, R0-R3= top elements of stack, R4=local variable 0). This hardware reuse contributes to the small size of the additional logic (12k gates) required to implement the Java machine, and keeps all of the state required by the Jazelle extension in ARM registers, In addition, it ensures compatibility with existing operating systems, interrupt handlers and exception code. ARM White Paper, Accelerating to Meet the Challenges of Embedded Java™ [TPLBN051478-TPLBN051482] ("ARM White Paper") at 3 [TPLBN051480].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

Figure 2-1 shows a top-level diagram of the Cortex-A9 processor.



Cortex-A9 TRM at 2-2 (33) [TPLBN034890].

11.3.1 Cortex-A9 specific events

Table 11-7 shows the Cortex-A9 specific events. In the value column of Table 11-7 Precise means the event is counted precisely. Events related to stalls and speculative instructions appear as Approximate entries in this column. s

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

0x70	Main execution unit instructions Counts the number of instructions being executed in the main execution pipeline of the processor, the multiply pipeline and arithmetic logic unit pipeline. The counted instructions are still speculative.	Approximate
0x71	Second execution unit instructions Counts the number of instructions being executed in the processor second execution pipeline (ALU). The counted instructions are still speculative.	Approximate

Cortex-A9 TRM at 11-7 – 11-8 (168-69).

In addition, the ARM architecture gives you:

- control over both the *Arithmetic Logic Unit* (ALU) and shifter in every data-processing instruction to maximize the use of an ALU and a shifter

ARM Architecture Reference Manual at A1-2 [PIC00005214].

The Arithmetic Logic Unit (ALU) derives its two inputs from the 'Top item' (Rn) and the 'Next item' (Rm) of the 'Push Down Stack' from the 'General Purpose Registers' (the holding place for items placed there by stack operations) and directs its output (Rd) back to the 'General Purpose Registers'.

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

A8.6.2 ADC (register)

Add with Carry (register) adds a register value, the carry flag value, and an optionally-shifted register value, and writes the result to the destination register. It can optionally update the condition flags based on the result.

Encoding T1 ARMv4T, ARMv5T*, ARMv6*, ARMv7

```
ADCS <Rdn>,<Rm>                                   Outside IT block.
ADC<c> <Rdn>,<Rm>                                 Inside IT block.
```

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	0	0	1	0	1	Rm			Rdn		

```
d = UInt(Rdn);  n = UInt(Rdn);  m = UInt(Rm);  setflags = !InITBlock();
(shift_t, shift_n) = (SRType_LSL, 0);
```

Encoding T2 ARMv6T2, ARMv7

```
ADC{S}<c>.W <Rd>,<Rn>,<Rm>{,<shift>}
```

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	0	1	0	1	1	0	1	0	S	Rn				(0)	imm3			Rd				imm2		type		Rm			

```
d = UInt(Rd);  n = UInt(Rn);  m = UInt(Rm);  setflags = (S == '1');
(shift_t, shift_n) = DecodeImmShift(type, imm3:imm2);
if BadReg(d) || BadReg(n) || BadReg(m) then UNPREDICTABLE;
```

Encoding A1 ARMv4*, ARMv5T*, ARMv6*, ARMv7

```
ADC{S}<c> <Rd>,<Rn>,<Rm>{,<shift>}
```

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cond				0	0	0	0	1	0	1	S	Rn				Rd				imm5				type		0	Rm				

```
if Rd == '1111' && S == '1' then SEE SUBS PC, LR and related instructions;
d = UInt(Rd);  n = UInt(Rn);  m = UInt(Rm);  setflags = (S == '1');
(shift_t, shift_n) = DecodeImmShift(type, imm5);
```

ARMv7 Reference Manual at A8-16 (328) [TPLBN049616].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

Assembler syntax

ADC{S}<c><q> {<Rd>,} <Rn>, <Rm> {,<shift>}

where:

S	If S is present, the instruction updates the flags. Otherwise, the flags are not updated.
<c><q>	See *Standard assembler syntax fields* on page A8-7.
<Rd>	The destination register.
<Rn>	The first operand register.
<Rm>	The optionally shifted second operand register.
<shift>	The shift to apply to the value read from <Rm>. If present, encoding T1 is not permitted. If absent, no shift is applied and any encoding is permitted. *Shifts applied to a register* on page A8-10 describes the shifts and how they are encoded.

ARMv7 Reference Manual at A8-17 (329) [TPLBN049617].

The Register File includes register R13, also known as SP or the Stack pointer, which is a pointer to the active stack.

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

A2.3 ARM core registers

In the application level view, an ARM processor has:

- thirteen general-purpose32-bit registers, R0 to R12
- three 32-bit registers, R13 to R15, that sometimes or always have a special use.

Registers R13 to R15 are usually referred to by names that indicate their special uses:

SP, the Stack Pointer

Register R13 is used as a pointer to the active stack.

In Thumb code, most instructions cannot access SP. The only instructions that can access SP are those designed to use SP as a stack pointer.

The use of SP for any purpose other than as a stack pointer is deprecated.

Note

Using SP for any purpose other than as a stack pointer is likely to break the requirements of operating systems, debuggers, and other software systems, causing them to malfunction.

LR, the Link Register

Register R14 is used to store the return address from a subroutine. At other times, LR can be used for other purposes.

When a `BL` or `BLX` instruction performs a subroutine call, LR is set to the subroutine return address. To perform a subroutine return, copy LR back to the program counter. This is typically done in one of two ways, after entering the subroutine with a `BL` or `BLX` instruction:

- Return with a `BX LR` instruction.
- On subroutine entry, store LR to the stack with an instruction of the form:
  `PUSH {<registers>,LR}`
  and use a matching instruction to return:
  `POP {<registers>,PC}`

ThumbEE checks and handler calls use LR in a similar way. For details see Chapter A9 *ThumbEE*.

ARMv7 Reference Manual at A2-11 (43) [TPLBN049331]; *see also* Cortex-A9 TRM at 1-4 (21) [TPLBN034878], 2-2 (33) [TPLBN034890], 7-7 (121) [TPLBN034978].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

A6.1.3 Use of 0b1101 as a register specifier

R13 is defined in the Thumb instruction set so that its use is primarily as a stack pointer, and R13 is normally identified as SP in Thumb instructions. In 32-bit Thumb instructions, if you use R13 as a general-purpose register beyond the architecturally defined constraints described in this section, the results are UNPREDICTABLE.

The restrictions applicable to R13 are described in:

- *R13[1:0] definition*
- *32-bit Thumb instruction support for R13.*

See also *16-bit Thumb instruction support for R13* on page A6-5.

ARMv7 Reference Manual at A6-4 (242) [TPLBN049530].

The top two items in the stack are connected to provide inputs into the ALU by using a "POP".

A8.6.122 POP

Pop Multiple Registers loads multiple registers from the stack, loading from consecutive memory locations starting at the address in SP, and updates SP to point just above the loaded data.

ARMv7 Reference Manual at A8-246 (558) [TPLBN049846].

The output of the ALU is connected to the top of the stack and may be saved with a "PUSH."

A8.6.123 PUSH

Push Multiple Registers stores multiple registers to the stack, storing to consecutive memory locations ending just below the address in SP, and updates SP to point to the start of the stored data.

ARMv7 Reference Manual at A8-248 (560) [TPLBN049848].

In the alternative, ARM Cortex A9 processor also includes an operand stack for Jazelle DBX instructions.

A2.10.2 Jazelle direct bytecode execution support

From ARMv5TEJ, the architecture requires every system to include an implementation of the Jazelle extension. The Jazelle extension provides architectural support for hardware acceleration of bytecode execution by a *Java Virtual Machine* (JVM).

ARMv7 Reference Manual at A2-73 (105) [TPLBN049393].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

Reads

Reads are defined as memory operations that have the semantics of a load.

The memory accesses of the following instructions are reads:

- LDR, LDRB, LDRH, LDRSB, and LDRSH
- LDRT, LDRBT, LDRHT, LDRSBT, and LDRSHT
- LDREX, LDREXB, LDREXD, and LDREXH
- LDM, LDRD, POP, and RFE
- LDC, LDC2, VLDM, VLDR, VLD1, VLD2, VLD3, and VLD4
- the return of status values by STREX, STREXB, STREXD, and STREXH
- in the ARM instruction set only, SWP and SWPB
- in the Thumb instruction set only, TBB and TBH.

Hardware-accelerated opcode execution by the Jazelle extension can cause a number of reads to occur, according to the state of the operand stack and the implementation of the Jazelle hardware acceleration.

Writes

Writes are defined as memory operations that have the semantics of a store.

The memory accesses of the following instructions are Writes:

- STR, STRB, and STRH
- STRT, STRBT, and STRHT
- STREX, STREXB, STREXD, and STREXH
- STM, STRD, PUSH, and SRS
- STC, STC2, VSTM, VSTR, VST1, VST2, VST3, and VST4
- in the ARM instruction set only, SWP and SWPB.

Hardware-accelerated opcode execution by the Jazelle extension can cause a number of writes to occur, according to the state of the operand stack and the implementation of the Jazelle hardware acceleration.

ARMv7 Reference Manual at A3-42 (156) [TPLBN049444].

The operand stack contains a top item register and a next item register connected to provide inputs to the ALU and the output of the ALU is connected to the top item register. The first four elements of the stack are held in

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

the register file in registers R0-R3. The stack pointer is held in register R6.

> In Java State, the processor assigns several ARM registers to functions specific to the Java machine (e.g., R6= stack pointer, R0-R3= top elements of stack, R4=local variable 0). This hardware reuse contributes to the small size of the additional logic (12k gates) required to implement the Java machine, and keeps all of the state required by the Jazelle extension in ARM registers, In addition, it ensures compatibility with existing operating systems, interrupt handlers and exception code.

ARM White Paper at 3 [TPLBN051480].

3. **Samsung S3C6410 Accused Products:**

Datapath

The datapath consists of three pipelines:

- ALU, shift and Sat pipeline
- MAC pipeline
- load or store pipeline, see *Load Store Unit (LSU)* on page 1-11.

ALU, shift or Sat pipe

The ALU, shift and Sat pipeline executes most of the ALU operations, and includes a 32-bit barrel shifter. It consists of three pipeline stages:

Shift	The Shift stage contains the full barrel shifter. This stage performs all shifts, including those required by the LSU. The Shift stage implements saturating left shift that doubles the value of an operand and saturates it.

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

ARM11 TRM at 1-10 (36) [TPLBN035123].

ALU	The ALU stage performs all arithmetic and logic operations, and generates the condition codes for instructions that set these flags. The ALU stage consists of a logic unit, an arithmetic unit, and a flag generator. The pipeline logic evaluates the flag settings in parallel with the main adder in the ALU. The flag generator is enabled only on flag-setting operations. The ALU stage separates the carry chains of the main adder for 8-bit and 16-bit SIMD instructions.
Sat	The Sat stage implements the saturation logic required by the various classes of DSP instructions.

ARM11 TRM at 1-11 (37) [TPLBN035124].

Core	A core is that part of a processor that contains the ALU, the datapath, the general-purpose registers, the Program Counter, and the instruction decode and control circuitry.

ARM11 TRM at Glossary-8 (747) [TPLBN035834].

In addition, the ARM architecture gives you:

- control over both the *Arithmetic Logic Unit* (ALU) and shifter in every data-processing instruction to maximize the use of an ALU and a shifter

ARM Architecture Reference Manual at A1-2 [PIC00005214].

The Arithmetic Logic Unit (ALU) derives its two inputs from the 'Top item' (Rn) and the 'Next item' (Rm) of the 'Push Down Stack' from the 'General Purpose Registers' (the holding place for items placed there by stack operations) and directs its output (Rd) back to the 'General Purpose Registers'.

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

A6.7.1 ADC (register)

Add with Carry (register) adds a register value, the carry flag value, and an optionally-shifted register value, and writes the result to the destination register. It updates the condition flags based on the result.

Encoding T1 All versions of the Thumb instruction set.

ADCS <Rdn>,<Rm>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	0	0	1	0	1	Rm			Rdn		

ARMv6 Reference Manual at A6-106 (106) [TPLBN035952].

Assembler syntax

ADCS{<q>} {<Rd>,} <Rn>, <Rm>

where:

S The instruction updates the flags.

{<q>} See *Standard assembler syntax fields* on page A6-98.

<Rd> The destination register. If <Rd> is omitted, this register is the same as <Rn>.

<Rn> The register that contains the first operand.

<Rm> The register that is optionally shifted and used as the second operand.

ARMv6 Reference Manual at A6-106 (106) [TPLBN035952].

The Register File includes register R13, also known as SP or the Stack pointer, which is a pointer to the active stack.

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

A2.3.1 ARM core registers

There are thirteen general-purpose 32-bit registers, R0-R12, and an additional three 32-bit registers that have special names and usage models:

SP Stack Pointer, used a pointer to the active stack. For usage restrictions see *Use of 0b1101 as a register specifier* on page A5-83. This is preset to the top of the Main stack on reset. See *The SP registers* on page B1-211 for more information. SP is sometimes referred to as R13.

LR Link Register stores the Return Link. This is a value that relates to the return address from a subroutine that is entered using a Branch with Link instruction. The LR register is also updated on exception entry, see *Exception entry behavior* on page B1-224. LR is sometimes referred to as R14.

Note

LR can be used for other purposes when it is not required to support a return from a subroutine.

ARMv6 Reference Manual at A2-36 (36) [TPLBN035882].

A5.1.3 Use of 0b1101 as a register specifier

R13 is defined in the Thumb instruction set so that its use is primarily as a stack pointer, aligning R13 with the *ARM Architecture Procedure Call Standard* (AAPCS), the architecture usage model supported by the PUSH and POP instructions.

ARMv6 Reference Manual at A5-83 (83) [TPLBN035929].

The top two items in the stack are connected to provide inputs into the ALU by using a “POP”.

A6.7.49 POP

Pop Multiple Registers loads a subset, or possibly all, of the general-purpose registers R0-R7 and the PC from the stack.

ARMv6 Reference Manual at A6-165 (165) [TPLBN036011].

The output of the ALU is connected to the top of the stack and may be saved with a “PUSH.”

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

	A6.7.50 PUSH Push Multiple Registers stores a subset, or possibly all, of the general-purpose registers R0-R7 and the LR to the stack. ARMv6 Reference Manual at A6-167 (167) [TPLBN036013]. In the alternative, ARM Cortex A9 processor also includes an operand stack for Jazelle DBX instructions. **A2.10.2 Jazelle direct bytecode execution support** From ARMv5TEJ, the architecture requires every system to include an implementation of the Jazelle extension. The Jazelle extension provides architectural support for hardware acceleration of bytecode execution by a *Java Virtual Machine* (JVM). ARMv6 Reference Manual at A6-167 (167) [TPLBN036013]. **Reads** Reads are defined as memory operations that have the semantics of a load. For ARMv6-M and Thumb these are: • LDR{S}B, LDR{S}H, LDR • LDM, POP **Writes** Writes are defined as operations that have the semantics of a store. For ARMv6-M and Thumb these are: • STRB, STRH, STR • STM, PUSH ARMv6 Reference Manual at A3-58 (58) [TPLBN035904]. The operand stack contains a top item register and a next item register connected to provide inputs to the ALU and the output of the ALU is connected to the top item register. The first four elements of the stack are held in the register file in registers R0-R3. The stack pointer is held in register R6.

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

	In Java State, the processor assigns several ARM registers to functions specific to the Java machine (e.g., R6= stack pointer, R0-R3= top elements of stack, R4=local variable 0). This hardware reuse contributes to the small size of the additional logic (12k gates) required to implement the Java machine, and keeps all of the state required by the Jazelle extension in ARM registers, In addition, it ensures compatibility with existing operating systems, interrupt handlers and exception code. ARM White Paper at 3 [TPLBN051480]. Plaintiffs contend that this claim element is literally present as described above. In the event that this claim element is not found to be literally present, Plaintiffs contend that the above-identified stacks are equivalent to "a first push down stack connected to said arithmetic logic unit," and any differences are insubstantial. In particular, the stacks perform the same function (*i.e.*, input and output to the ALU), in substantially the same way (*i.e.*, by providing a last-in, first-out data structure), and have the same result (*i.e.*, the ALU performs operations on inputs and provides output).
[1h] said first push down stack including means for storing a top item connected to a first input of said arithmetic logic unit to provide the top item to the first input and means for storing a next item connected to	On information and belief, in each Accused Product, the first push down stack includes means for storing a top item connected to a first input of said arithmetic logic unit to provide the top item to the first input. On information and belief, each Accused Product has means for storing a next item connected to a second input of said arithmetic logic unit to provide the next item to the second input. The Arithmetic Logic Unit (ALU) derives its two inputs from the 'Top item' (Rn) and the 'Next item' (Rm) of the 'Push Down Stack' from the 'General Purpose Registers' (the holding place for items placed there by stack operations) and directs its output (Rd) back to the 'General Purpose Registers'. *See also* the evidence shown above for item 1g.

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

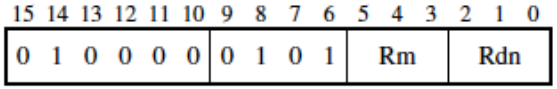
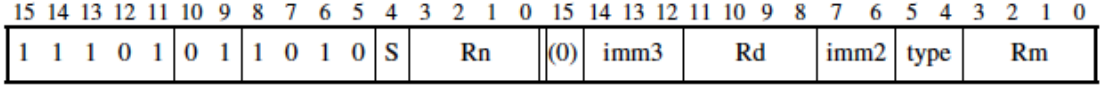
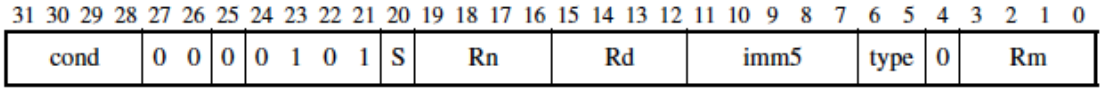
a second input of said arithmetic logic unit to provide the next item to the second input,	**A8.6.2 ADC (register)**
	Add with Carry (register) adds a register value, the carry flag value, and an optionally-shifted register value, and writes the result to the destination register. It can optionally update the condition flags based on the result.
	Encoding T1 ARMv4T, ARMv5T*, ARMv6*, ARMv7
	`ADCS <Rdn>,<Rm>` Outside IT block.
	`ADC<c> <Rdn>,<Rm>` Inside IT block.
	 
	
	`d = UInt(Rdn); n = UInt(Rdn); m = UInt(Rm); setflags = !InITBlock();`
	`(shift_t, shift_n) = (SRType_LSL, 0);`
	Encoding T2 ARMv6T2, ARMv7
	`ADC{S}<c>.W <Rd>,<Rn>,<Rm>{,<shift>}`
	 
	
	`d = UInt(Rd); n = UInt(Rn); m = UInt(Rm); setflags = (S == '1');`
	`(shift_t, shift_n) = DecodeImmShift(type, imm3:imm2);`
	`if BadReg(d) || BadReg(n) || BadReg(m) then UNPREDICTABLE;`
	Encoding A1 ARMv4*, ARMv5T*, ARMv6*, ARMv7
	`ADC{S}<c> <Rd>,<Rn>,<Rm>{,<shift>}`
	 
	
	`if Rd == '1111' && S == '1' then SEE SUBS PC, LR and related instructions;`
	`d = UInt(Rd); n = UInt(Rn); m = UInt(Rm); setflags = (S == '1');`
	`(shift_t, shift_n) = DecodeImmShift(type, imm5);`
	ARMv7 Reference Manual at A8-16 (328) [TPLBN049616]; *see also* ARMv6 Reference Manual at A6-106 (106) [TPLBN035952].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

Assembler syntax

```
ADC{S}<c><q>  {<Rd>,} <Rn>, <Rm> {,<shift>}
```

where:

S	If S is present, the instruction updates the flags. Otherwise, the flags are not updated.
<c><q>	See *Standard assembler syntax fields* on page A8-7.
<Rd>	The destination register.
<Rn>	The first operand register.
<Rm>	The optionally shifted second operand register.
<shift>	The shift to apply to the value read from <Rm>. If present, encoding T1 is not permitted. If absent, no shift is applied and any encoding is permitted. *Shifts applied to a register* on page A8-10 describes the shifts and how they are encoded.

ARMv7 Reference Manual at A8-17 (329) [TPLBN049617]; *see also* ARMv6 Reference Manual at A6-106 (106) [TPLBN035952].

On information and belief, the operand stack contains a top item register and a next item register connected to provide inputs to the ALU and the output of the ALU is connected to the top item register.

The top item in the stack is connected to provide input to an internal data bus by using a "POP" which loads it into a register.

A8.6.122 POP

Pop Multiple Registers loads multiple registers from the stack, loading from consecutive memory locations starting at the address in SP, and updates SP to point just above the loaded data.

ARMv7 Reference Manual at A8-246 (558) [TPLBN049846]; *see also* ARMv6 Reference Manual at A6-165 (165) [TPLBN036011].

The output of the ALU is connected to the top of the stack and may be saved with a "PUSH."

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

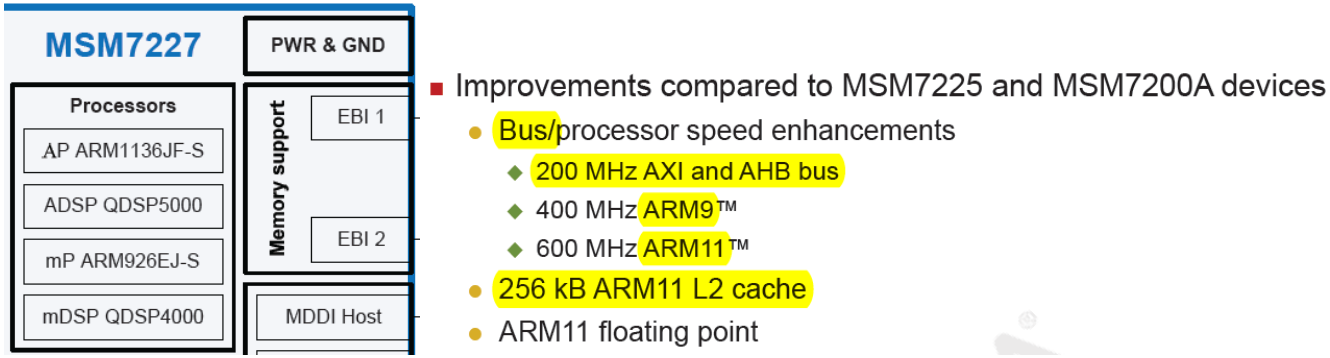
A8.6.123 PUSH

> Push Multiple Registers stores multiple registers to the stack, storing to consecutive memory locations ending just below the address in SP, and updates SP to point to the start of the stored data.

ARMv7 Reference Manual at A8-248 (560) [TPLBN049848]; *see also* ARMv6 Reference Manual at A6-167 (167) [TPLBN036013].

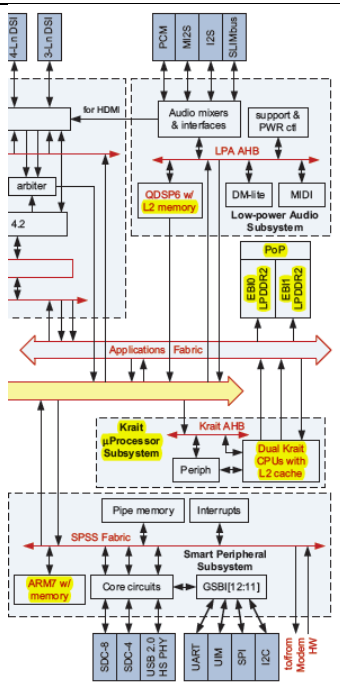
The top item of the stack is also located in the data cache within the "memory system."

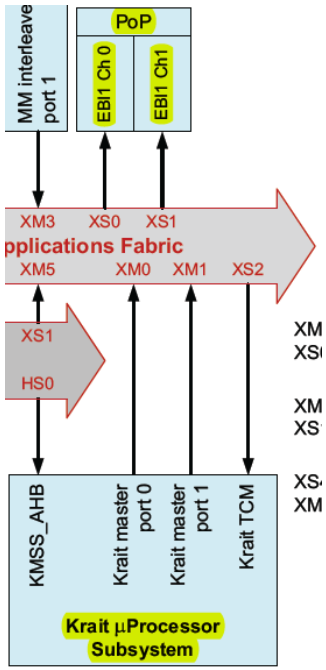
The register file is connected through the memory system which is bidirectionally connected to the data bus of each of the Qualcomm, OMAP 3621, OMAP4470, OMAP4430, and Samsung S3C6410 processors, as shown above.

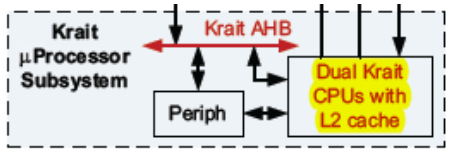


MSM7227 Chipset Training at 5, 7. *See also* the ARM core documentation cited below.

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung







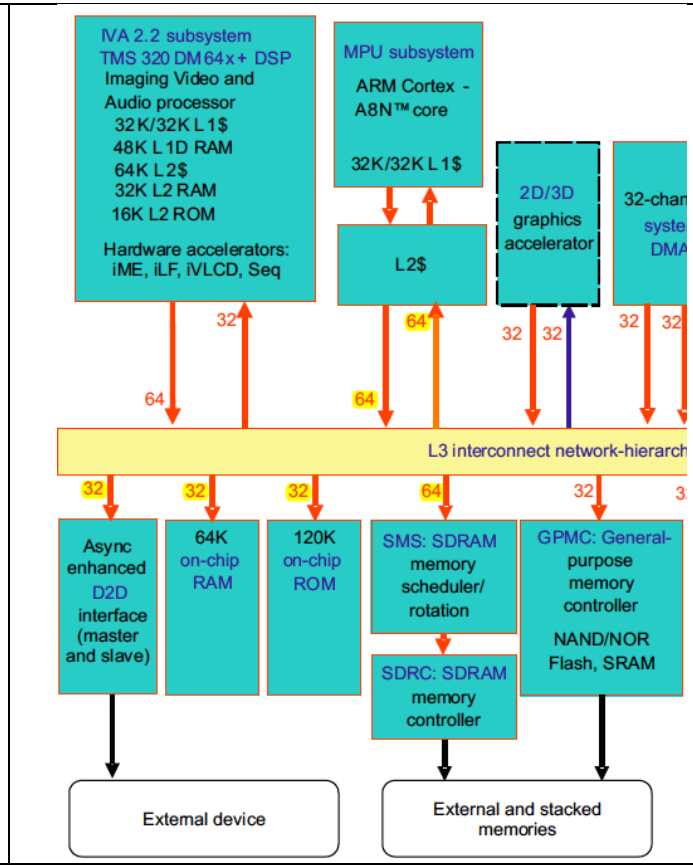
Major MSM8960 functional blocks (left) are shared by its internal systems, subsystem, and busses – the 'architecture' (right)

Key DDR memory controller features

- Supports DDR on address bus
- Two major clock domains (AXI slave and DDR controller)
- Multiple AXI-to-DDR clock modes
  – Synchronous (1:1), iso-synchronous (1:2), asynchronous
- Highly flexible with many configurable parameters, including interface timing
- Flexible memory page management with various page open/close policies
- Out-of-order command execution and read data return
- Multiple high-priority tiers for sophisticated handling of priority requests

MSM8960 Design Guidelines at 38, 44, 51, 94. *See also* the ARM core documentation cited below.

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung



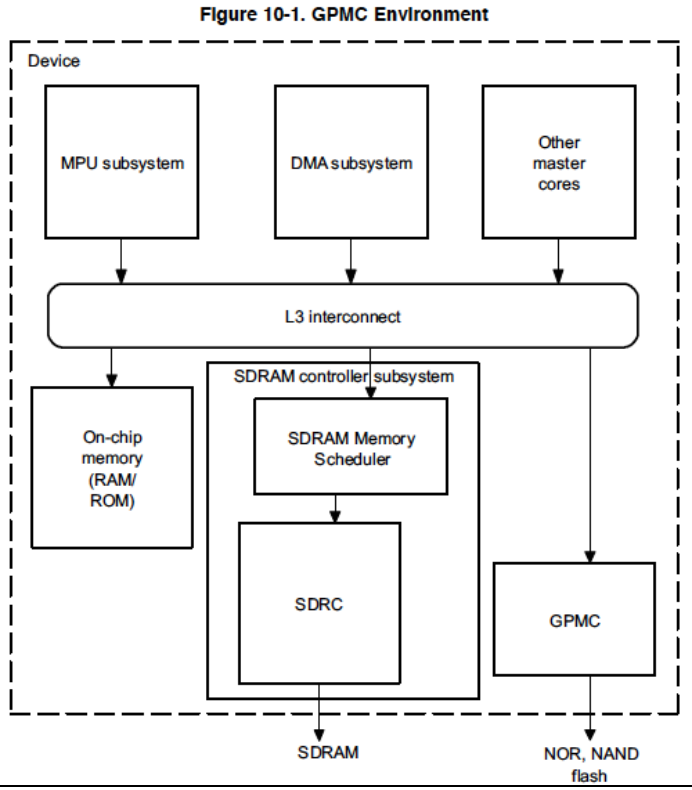
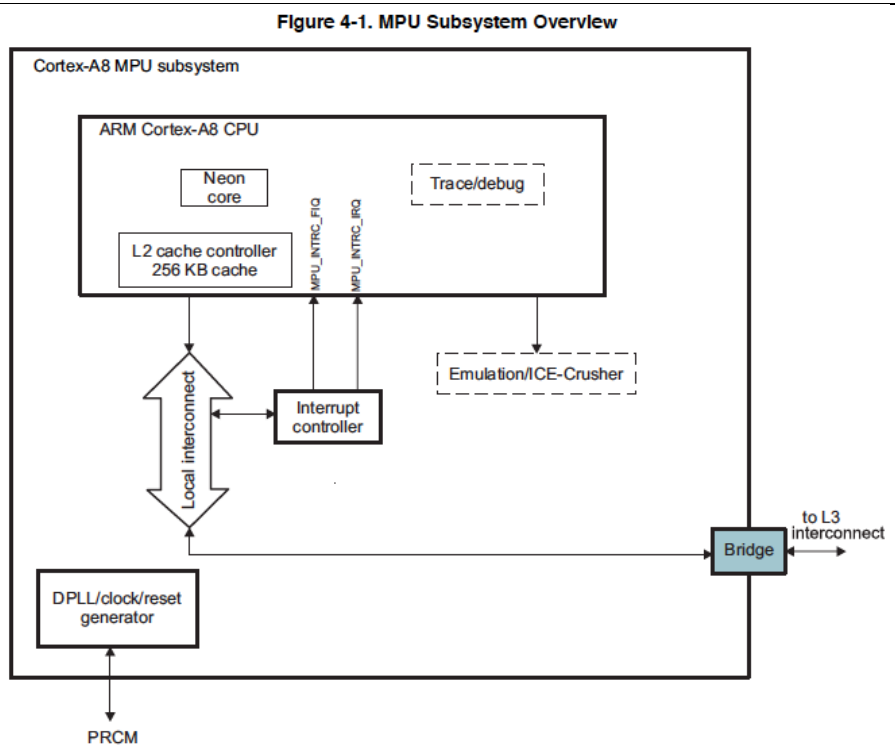


EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung



OMAP36xx TRM at 191 [TPLBN004904], 679 [TPLBN005392], 2129 [TPLBN006842].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

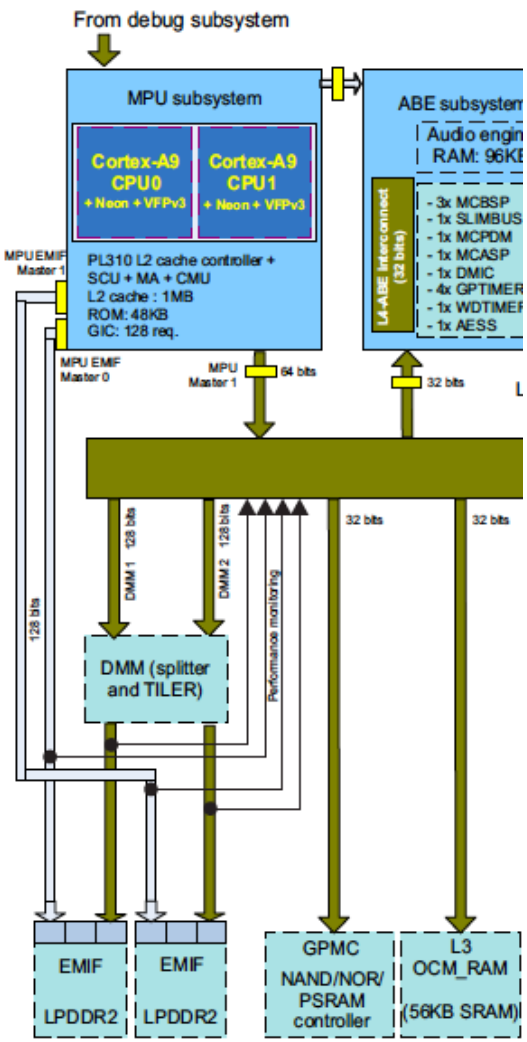


Figure 16-1. Memory Subsystem Functional Diagram

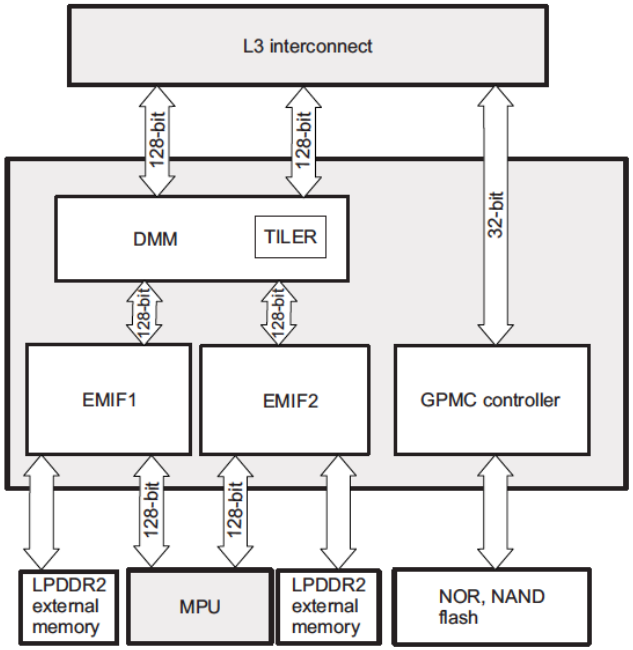
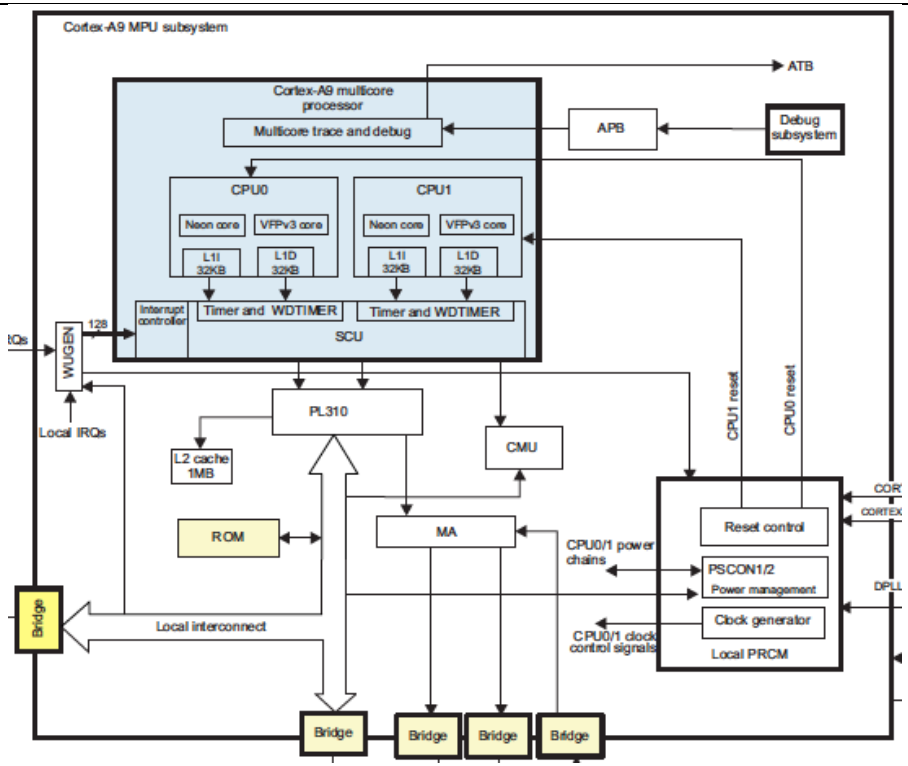


EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung



OMAP4470 TRM at 276 [TPLBN021476], 1090 [TPLBN022290], 3280 [TPLBN024480]; *see also* OMAP4430 TRM at 275 [TPLBN008774], 1077 [TPLBN009576], 3207 [TPLBN011706].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

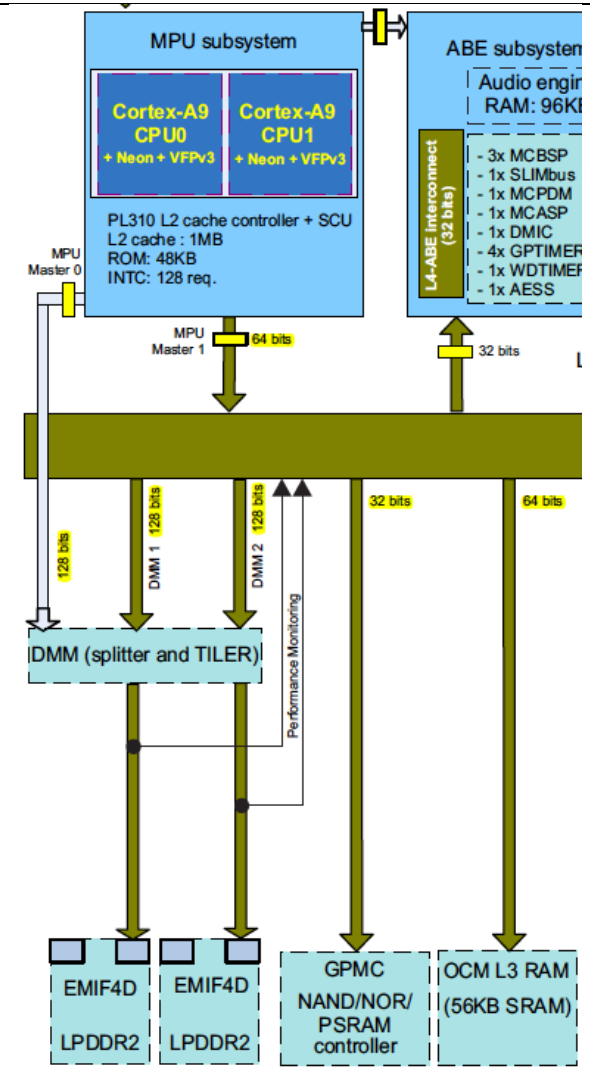
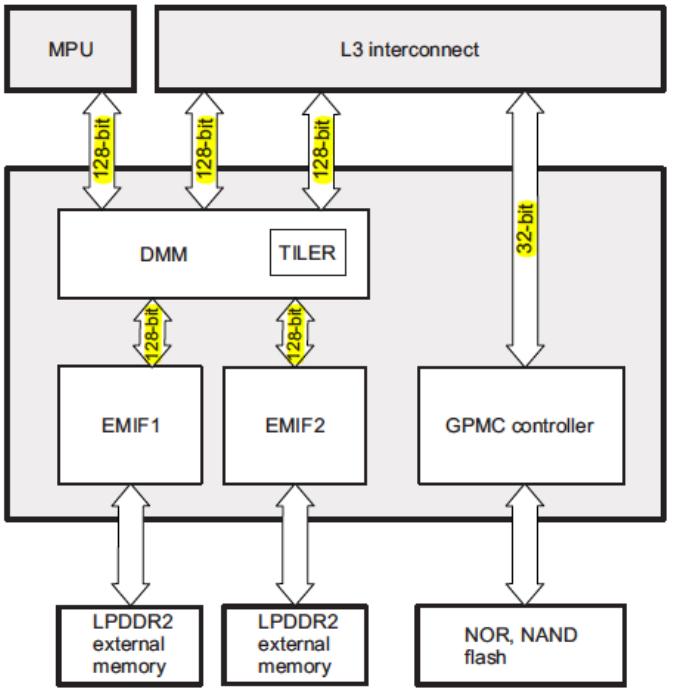
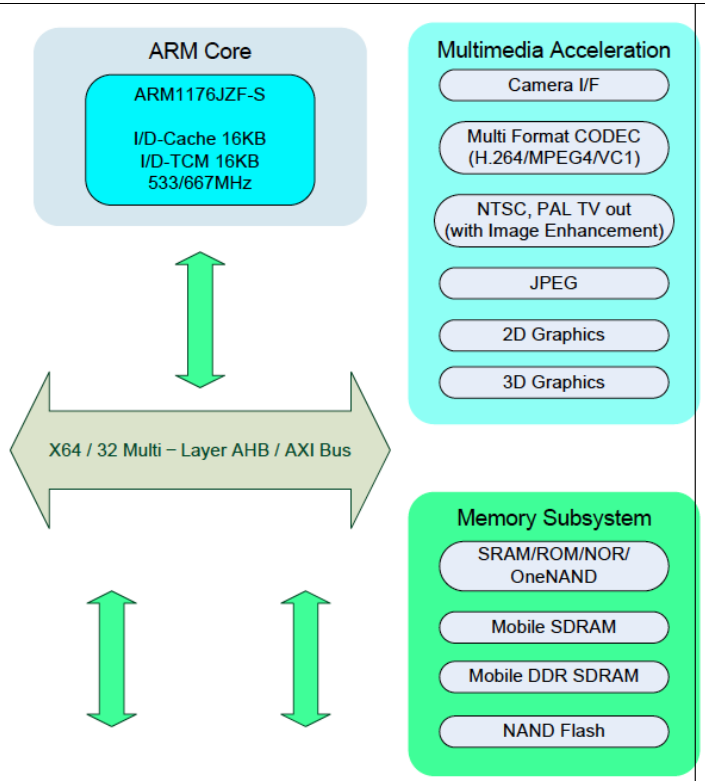


Figure 15-1. Memory Subsystem Functional Diagram



OMAP4430 TRM at 275 [TPLBN008774], 3207 [TPLBN011706]; *see also* OMAP4460 TRM at 263, 2982; OMAP4470 TRM at 276 [TPLBN021476], 1090 [TPLBN022290], 3280 [TPLBN024480]; OMAP4430 TRM at 275 [TPLBN008774], 1077 [TPLBN009576], 3207 [TPLBN011706].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung



S3C6410X User Manual at 1-2 (61) [TPLBN002468].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

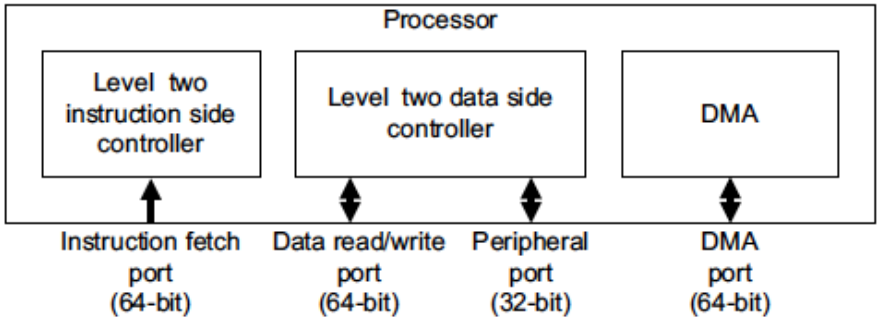
	 **Figure 8-1 Level two interconnect interfaces** ARM11 TRM at 8-2 (389) [TPLBN035476]. As discussed in connection with claim element 1g, Plaintiffs contend that this claim element is literally present as described above. In the event that this claim element is not found to be literally present, Plaintiffs contend that the above-identified stacks are equivalent to "first push down stack including means for storing a top item connected to a first input of said arithmetic logic unit to provide the top item to the first input and means for storing a next item connected to a second input of said arithmetic logic unit to provide the next item to the second input" and any differences are insubstantial. In particular, the stacks perform the same function (*i.e.*, input and output to the ALU), in substantially the same way (*i.e.*, by providing a last-in, first-out data structure), and have the same result (*i.e.*, the ALU performs operations on inputs and provides output). *See* element 1g, above.
[1i] a remainder of said first push down stack being connected to said means for storing a next item to receive the next item from said means for storing a	On information and belief, each Accused Product has a remainder of said first push down stack being connected to said means for storing a next item to receive the next item from said means for storing a next item when pushed down in said push down stack. As discussed above in connection with element 1g, the Arithmetic Logic Unit (ALU) derives its two inputs from the 'Top item' (Rn) and the 'Next item' (Rm) of the 'Push Down Stack' from the 'General Purpose Registers' (the holding place for items placed there by stack operations) and directs its output (Rd) back to the 'General Purpose Registers'. *See* element 1g, above.

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

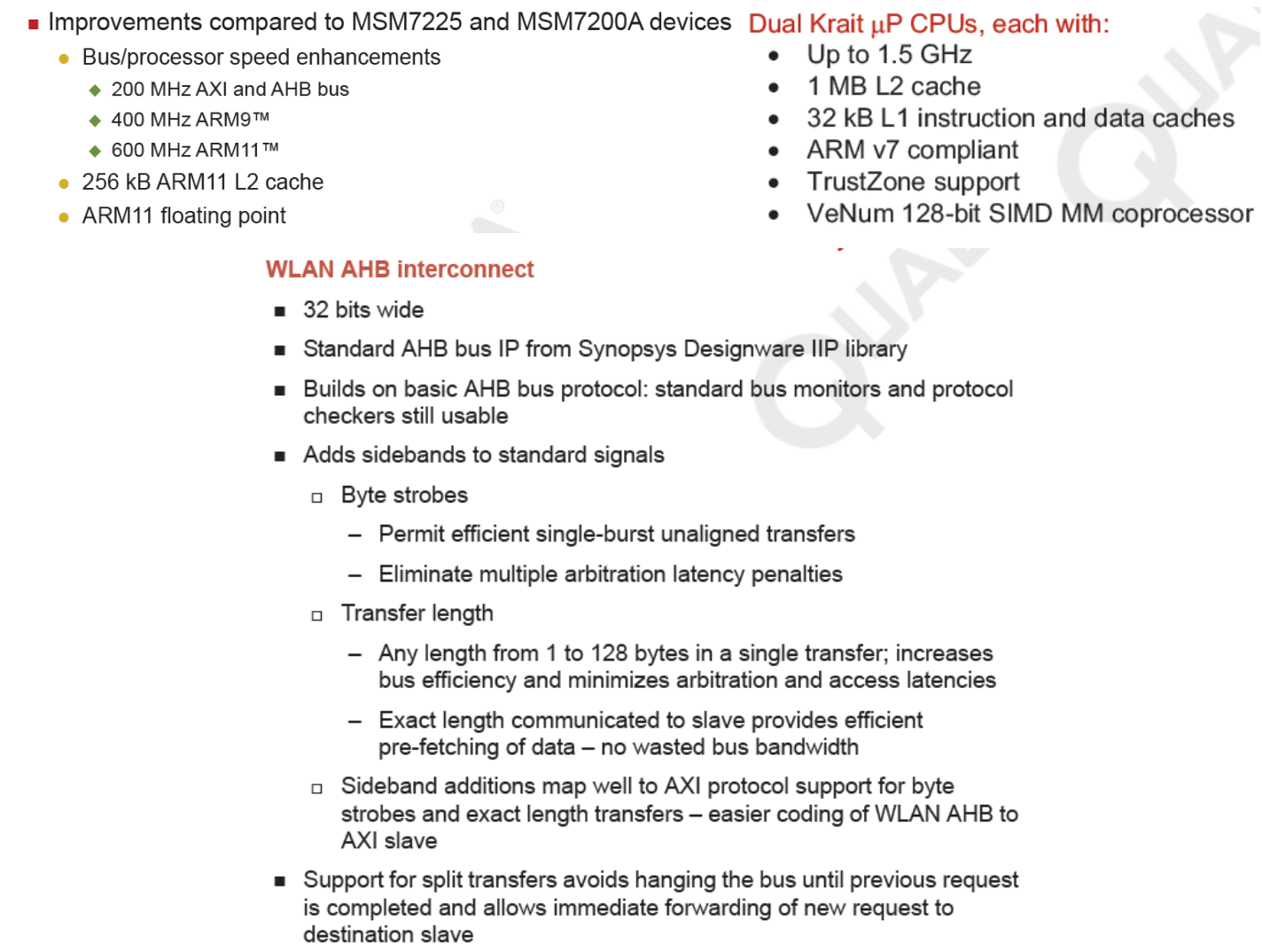
next item when pushed down in said push down stack,	As to Accused Products with Qualcomm processors, *see e.g.*:  ■ Improvements compared to MSM7225 and MSM7200A devices ● Bus/processor speed enhancements ◆ 200 MHz AXI and AHB bus ◆ 400 MHz ARM9™ ◆ 600 MHz ARM11™ ● 256 kB ARM11 L2 cache ● ARM11 floating point Dual Krait µP CPUs, each with: • Up to 1.5 GHz • 1 MB L2 cache • 32 kB L1 instruction and data caches • ARM v7 compliant • TrustZone support • VeNum 128-bit SIMD MM coprocessor WLAN AHB interconnect ■ 32 bits wide ■ Standard AHB bus IP from Synopsys Designware IIP library ■ Builds on basic AHB bus protocol: standard bus monitors and protocol checkers still usable ■ Adds sidebands to standard signals □ Byte strobes – Permit efficient single-burst unaligned transfers – Eliminate multiple arbitration latency penalties □ Transfer length – Any length from 1 to 128 bytes in a single transfer; increases bus efficiency and minimizes arbitration and access latencies – Exact length communicated to slave provides efficient pre-fetching of data – no wasted bus bandwidth □ Sideband additions map well to AXI protocol support for byte strobes and exact length transfers – easier coding of WLAN AHB to AXI slave ■ Support for split transfers avoids hanging the bus until previous request is completed and allows immediate forwarding of new request to destination slave MSM7227 Chipset Training at 7; *see also* MSM8960 Design Guidelines at 454, 86, 88; http://www.anandtech.com/show/4940/qualcomm-new-snapdragon-s4-msm8960-krait-architecture

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

(showing fetch and decode stages for the Krait core; "The architecture can fetch and decode three instructions per clock. The decoders are equally capable of decoding any ARMv7-A instructions."); *id* (showing decoding of multiple instructions in parallel). *See also* the ARM core documentation cited below.

Each ARM processor has a set of multiple registers with 32 bits:

2.13 Registers

The processor has a total of 40 registers:

- 33 general-purpose 32-bit registers
- seven 32-bit status registers.

These registers are not all accessible at the same time. The processor state and mode of operation determine the registers that are available to the programmer.

2.13.1 The state register set

In ARM state, 16 data registers and one or two status registers are accessible at any time. In privileged modes, mode-specific banked registers become available. Figure 2-10 on page 2-19 shows the registers that are available in each mode.

Thumb and ThumbEE state give access to the same set of registers as ARM state. However, the 16-bit instructions provide only limited access to some of the registers. No such limitations exist for 32-bit Thumb-2 and ThumbEE instructions.

Registers r0 through r13 are general-purpose registers used to hold either data or address values.

Cortex-A8 TRM at 2-18 (58) [TPLBN034335]; *see also* ARM11 TRM at 2-18 (91) [TPLBN035178].

The scheme maps the 32 ARM architectural registers to a pool of 56 physical 32-bit registers, and renames the flags (N, Z, C, V, Q, and GE) of the CPSR using a dedicated pool of eight physical 9-bit registers.

Cortex-A9 TRM at 2-2 (33) [TPLBN034890].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

Figure 2-11 shows an alternative view of the ARM registers.

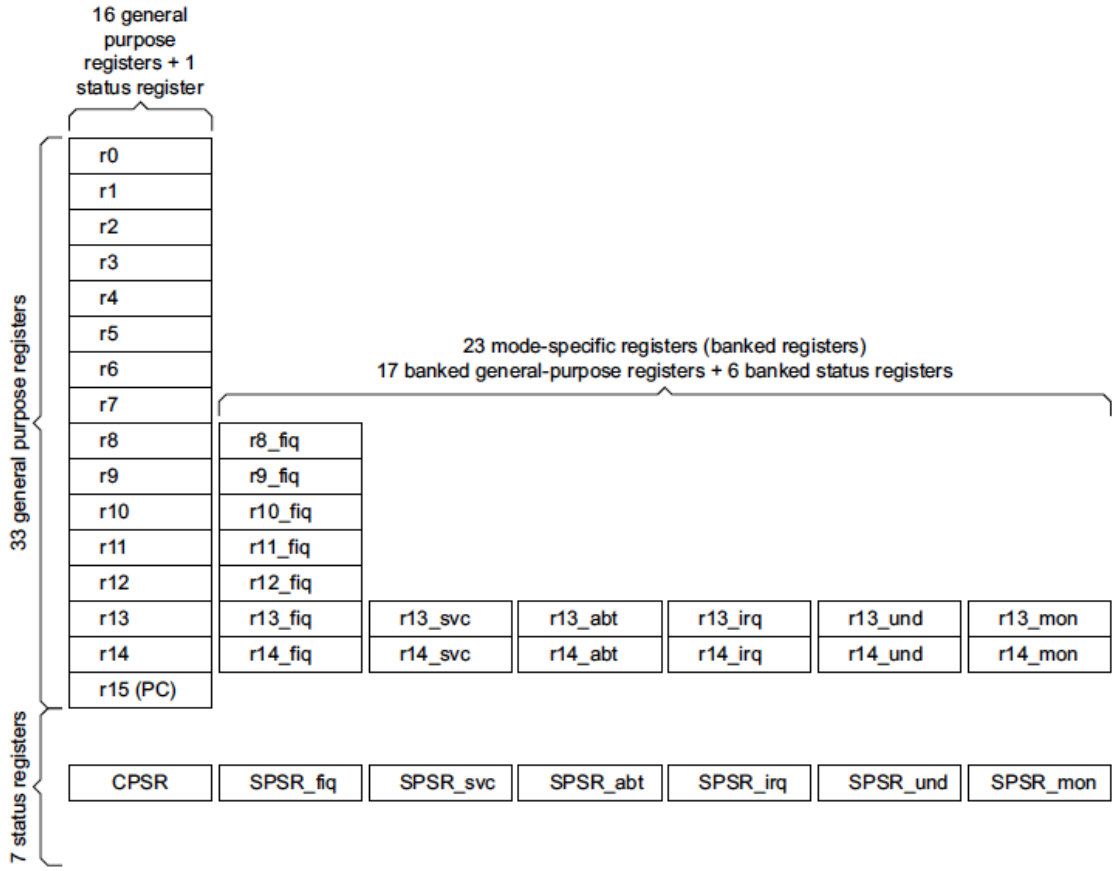


Figure 2-11 Processor register set showing banked registers

Cortex-A8 TRM at 2-20 (60) [TPLBN034337]; *see also* ARM11 TRM at 2-21 (94) [TPLBN035181]; A9 TRM at 7-7 (121) [TPLBN034978].

As discussed in connection with claim element 1g and 1h, Plaintiffs contend that this claim element is literally present as described above; however, for the reasons stated above, in the event that this claim element is not found to be literally present, Plaintiffs contend that the above-identified stacks are equivalent to this claim

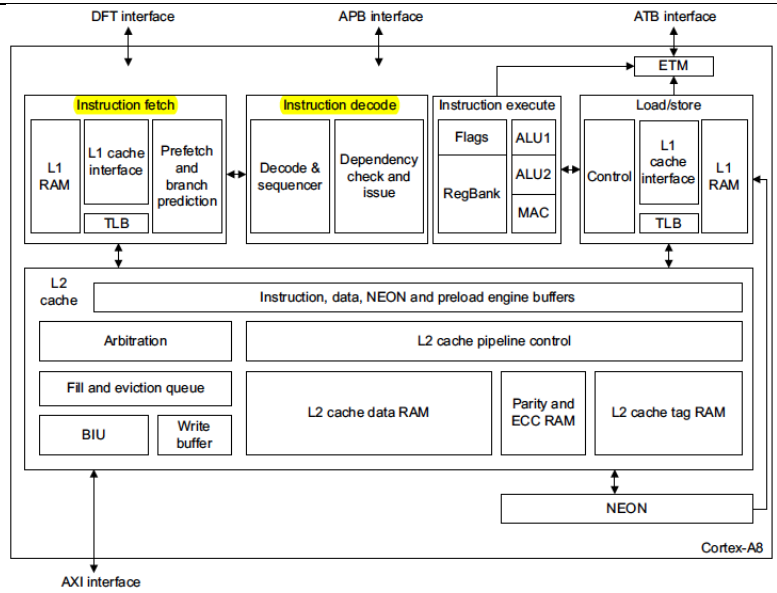
EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

	element. *See* elements 1g and 1h, above.
[1j] said arithmetic logic unit having an output connected to said means for storing a top item;	On information and belief, in each Accused Product, said arithmetic logic unit has an output connected to said means for storing a top item. As discussed above in connection with element 1g, the Arithmetic Logic Unit (ALU) derives its two inputs from the 'Top item' (Rn) and the 'Next item' (Rm) of the 'Push Down Stack' from the 'General Purpose Registers' (the holding place for items placed there by stack operations) and directs its output (Rd) back to the 'General Purpose Registers'. *See* element 1g, above; *see also* elements 1h, and 1i, above. The output is stored in these "General Purpose Registers." *See also* http://www.anandtech.com/show/4940/qualcomm-new-snapdragon-s4-msm8960-krait-architecture (showing fetch and decode stages for the Krait core; "The architecture can fetch and decode three instructions per clock. The decoders are equally capable of decoding any ARMv7-A instructions."); *id* (showing decoding of multiple instructions in parallel). *See also* the ARM core documentation cited herein.
[1k] wherein the microprocessor system comprises an instruction register configured to store the multiple sequential instructions and from which instructions are accessed and decoded; and	On information and belief, in each Accused Product, the microprocessor system comprises an instruction register configured to store the multiple sequential instructions and from which instructions are accessed and decoded.

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

	1.3.1 MPU Subsystem The MPU subsystem integrates the following modules: • ARM® subchip: – ARM Cortex-A8 core – ARM Version 7 ISA™: Standard ARM instruction set plus Thumb®-2, Jazelle® RCT Java accelerator, and media extensions – Neon™ SIMD coprocessor (VFP lite plus media streaming instructions) – Cache memories: • Level 1 (L1): 32-KB instruction and 32-KB data–4-way set associative cache, 64 bytes/line • Level 2 (L2): Up to 256KB. • Interrupt controller (MPU INTC) of 96 synchronous interrupt lines • Asynchronous interface with core logic • Debug, trace, and emulation features: ICECrusher™, ETM™, and ETB™ modules OMAP36xx TRM at 191 [TPLBN004904], 679 [TPLBN005392]; *see also* ARMv7 Reference Manual at A3-3 (117) [TPLBN049405] re: "prefetch[ing] instructions," A3-28 (142) [TPLBN049430] re: "instruction fetches"; *see also* OMAP4470 TRM at 277 [TPLBN021477] (Cortex-A9); OMAP4430 TRM at 276 [TPLBN008775] (Cortex-A9); S3C6410X User Manual at 1-4 (63) [TPLBN002470] (ARM1176JZF-S). **4.2.2.1 ARM Overview** The ARM Cortex-A8 processor incorporates the technologies available in the ARMv7 architecture. These technologies include NEON for media and signal processing and Jazelle® Runtime Compilation Target (RCT) for acceleration of realtime compilers, Thumb-2 technology for code density and the VFPv3 floating point architecture. For details, see the ARM Cortex-A8 Technical Reference Manual. OMAP36xx TRM at 685 [TPLBN005398]. ***4.2.2.2.1 ARM Cortex-A8 Instruction, Data, and Private Peripheral Port*** The CPU bus interface to the local interconnect is the main interface to the ARM system bus. It performs L2 cache fills and non-cacheable accesses for both instructions and data. The bus interface supports 64-bit wide input and output data buses. OMAP36xx TRM at 685 [TPLBN005398].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung



1.3 Components of the processor

The main components of the processor are:

- *Instruction fetch*
- *Instruction decode* on page 1-5
- *Instruction execute* on page 1-5
- *Load/store* on page 1-5
- *L2 cache* on page 1-5
- *NEON* on page 1-6
- *ETM* on page 1-6.

Cortex-A8 TRM at 1-4 (29) [TPLBN034306].

1.3.1 Instruction fetch

The instruction fetch unit predicts the instruction stream, fetches instructions from the L1 instruction cache, and places the fetched instructions into a buffer for consumption by the decode pipeline. The instruction fetch unit also includes the L1 instruction cache.

Cortex-A8 TRM at 1-4 (29) [TPLBN034306].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

1.3.2 Instruction decode

The instruction decode unit decodes and sequences all ARM and Thumb-2 instructions including debug control coprocessor, CP14, instructions and system control coprocessor, CP15, instructions. See Chapter 12 *Debug* for information on the CP14 coprocessor and Chapter 3 *System Control Coprocessor* for information on the CP15 coprocessor.

The instruction decode unit handles the sequencing of:

- exceptions
- debug events
- reset initialization
- *Memory Built-In Self Test* (MBIST)
- wait-for-interrupt
- other unusual events.

Cortex-A8 TRM at 1-5 (30) [TPLBN034307].

1.3.1 Cortex-A9 MPU Subsystem Description

The Cortex-A9 MPU subsystem integrates the following submodules:

- ARM Cortex-A9 MPCore
  - Two ARM Cortex-A9 central processing units (CPUs)
  - ARM Version 7 ISA™: Standard ARM instruction set plus Thumb®-2, Jazelle® RCT and Jazelle DBX Java™ accelerators
  - Neon™ SIMD coprocessor and VFPv3 per CPU
  - Interrupt controller (Cortex-A9 MPU INTC) with up to 128 interrupt requests
  - One general-purpose timer and one watchdog timer per CPU
  - Debug and trace features
  - 32-KB instruction and 32-KB data level 1 (L1) caches per CPU
- Shared 1-MB level 2 (L2) cache
- 48 KB bootable ROM
- Local power, reset, and clock managment (PRCM) module
- Emulation features
- Digital phase-locked loop (DPLL)

OMAP4430 TRM at 276 [TPLBN008775]; *see also* ARMv7 Reference Manual at A3-3 (117) [TPLBN049405]

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

	re: "prefetch[ing] instructions," A3-28 (142) [TPLBN049430] re: "instruction fetches". ***4.3.1 Cortex-A9 MPU Subsystem Block Diagram*** The Cortex-A9 MPU subsystem integrates the following group of submodules: • Two ARM Cortex-A9 CPUs. Each CPU contains: – ARM version 7 ISA™: Standard ARM instruction set plus Thumb®-2 , Jazelle® RCT and Jazelle DBX Java™ accelerator – Neon SIMD coprocessor and VFPv3 • INTC: Handles module interrupts (For details, see Chapter 17, *Interrupt Controller.*) • PL310 L2 cache controller (revision r2p0) with 1MB cache and two 64-bit slave and two 64-bit master ports (For details about L2 cache controller, see the ARM PL310 Cache Controller TRM, available at infocenter.arm.com/help/index.jsp). • Local interconnect: Connects the ARM Cortex-A9 multicore processor to the level 3 (L3) interconnect, dynamic memory manager, ABE interconnect, local PRCM, PL310 L2 cache controller, on-chip ROM memory, and the WUGEN. • Power, clock and reset manager • On-chip ROM memory – CPU0 (the master CPU) can boot from this memory. The ROM memory size is 48KB, the address range is from 0x4002 8000 to 0x4003 3FFF. For more information, see Chapter 27, *Initialization.* • WUGEN: Responsible for waking up the CPUs, used by the ROM code and OS during SMP boot. Two internal memory-mapped registers, AUX_CORE_BOOT_0 and AUX_CORE_BOOT_1, are available to the OS for communicating start-up information. For more information, see Chapter 27, *Initialization.* • Standby controllers – handle the power transitions inside the Cortex-A9 MPU subsystem. • SCU for L1 cache coherency • One timer and one watchdog unit per core • Caches memories: – 32-KB L1 instruction and 32-KB data caches – 4-way associative on each core – 1-MB L2 unified cache – 16-way associative OMAP4430 TRM at 1084 [TPLBN009583].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

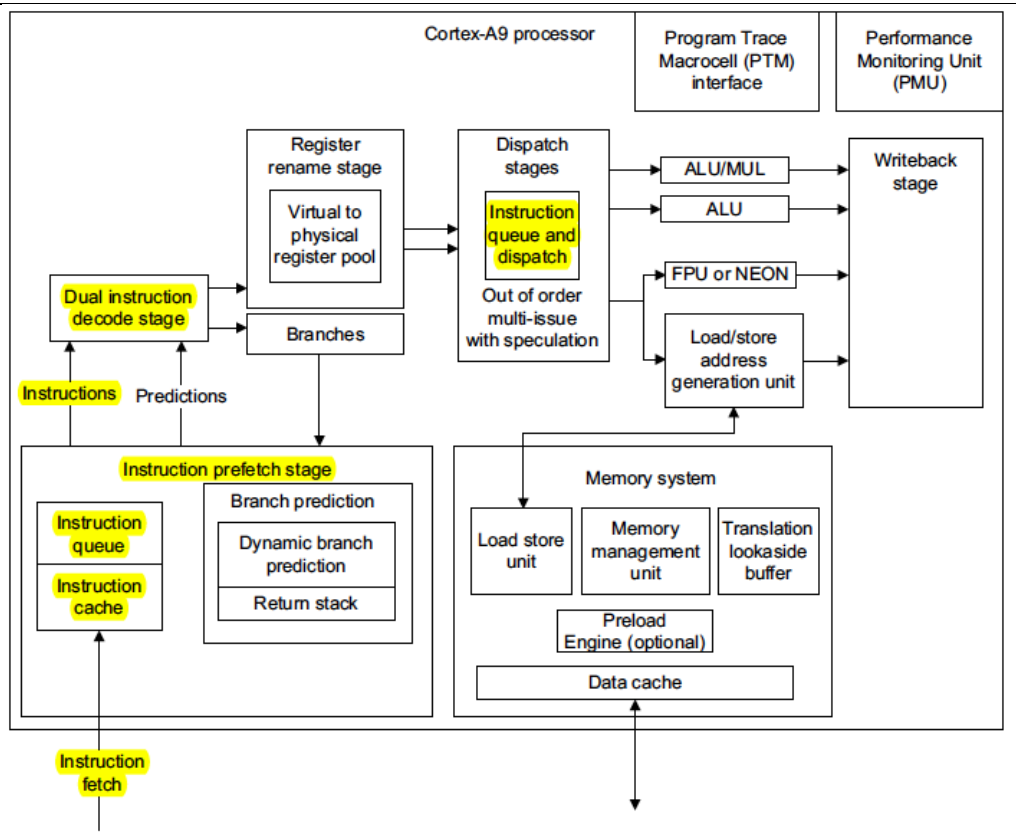


Figure 2-1 Cortex-A9 processor top-level diagram

Cortex-A9 TRM at 2-2 (33) [TPLBN034890].

2.1.2 Instruction queue

In the instruction queue small loop mode provides low power operation while executing small instruction loops. See *Energy efficiency features* on page 2-10.

Cortex-A9 TRM at 2-2 (33) [TPLBN034890]; ARM11 TRM at 11-12 (457) [TPLBN035544].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

	2.1.3 Dynamic branch prediction The Prefetch Unit implements two-level dynamic branch prediction with a *Global History Buffer* (GHB), a *Branch Target Address Cache* (BTAC) and a return stack. See *About the L1 instruction side memory system* on page 7-5. Cortex-A9 TRM at 2-3 (34) [TPLBN034891]; ARM11 TRM at 1-11 (37) [TPLBN035124]. **2.4.1 Energy efficiency features** The features of the Cortex-A9 processor that improve energy efficiency include: • accurate branch and return prediction, reducing the number of incorrect instruction fetch and decode operations • the use of physically addressed caches, reducing the number of cache flushes and refills, saving energy in the system • the use of micro TLBs reduces the power consumed in translation and protection look-ups for each cycle • caches that use sequential access information to reduce the number of accesses to the tag RAMs and to unnecessary accesses to data RAMs • instruction loops that are smaller than 64 bytes often complete without additional instruction cache accesses, so lowering power consumption. Cortex-A9 TRM at 2-10 (41) [TPLBN034898]; ARM11 TRM at 1-23 (49) [TPLBN035136].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

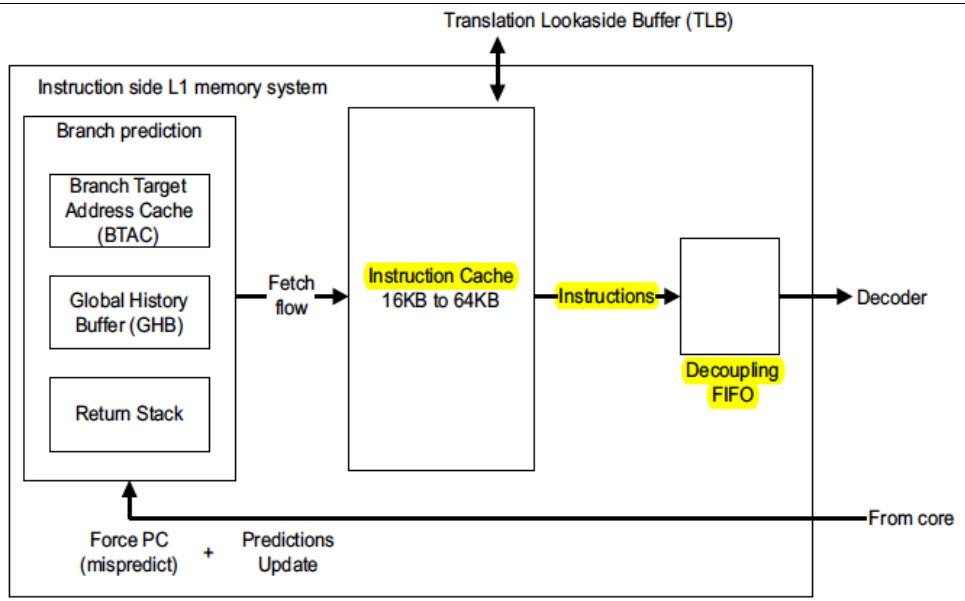


Figure 7-1 Branch prediction and instruction cache

Cortex-A9 TRM at 7-5 (119) [TPLBN034976]; *see also* Cortex-A8 TRM at 1-4 (29) [TPLBN034306]; ARM11 TRM at 1-26 (52) [TPLBN035139].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

Cache features

The Cortex-A9 processor has separate instruction and data caches. The caches have the following features:

- Each cache can be disabled independently. See *System Control Register* on page 4-15.
- Cache replacement policy is either pseudo round-robin or pseudo random.
- Both caches are 4-way set-associative.
- The cache line length is eight words.
- On a cache miss, critical word first filling of the cache is performed.
- You can configure the instruction and data caches independently during implementation to sizes of 16KB, 32KB, or 64KB.
- To reduce power consumption, the number of full cache reads is reduced by taking advantage of the sequential nature of many cache operations. If a cache read is sequential to the previous cache read, and the read is within the same cache line, only the data RAM set that was previously read is accessed.

Instruction cache features

The instruction cache is virtually indexed and physically tagged.

Cortex-A9 TRM at 7-2 (116) [TPLBN034898]; *see* ARM11 TRM at 1-8, 1-11, 1-16 (34, 37, 42) [TPLBN035121, TPLBN035124, TPLBN035129].

Prefetching	In pipelined processors, the process of fetching instructions from memory to fill up the pipeline before the preceding instructions have finished executing. Prefetching an instruction does not mean that the instruction must be executed.

Cortex-A9 TRM at Glossary-12 (226) [TPLBN035083]; ARM11 TRM at Glossary-15 (754) [TPLBN035841].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

A3.9.2 Memory hierarchy

Memory close to a processor has very low latency, but is limited in size and expensive to implement. Further from the processor it is easier to implement larger blocks of memory but these have increased latency. To optimize overall performance, an ARMv7 memory system can include multiple levels of cache in a hierarchical memory system. Figure A3-5 shows such a system, in an ARMv7-A implementation of a VMSA, supporting virtual addressing.

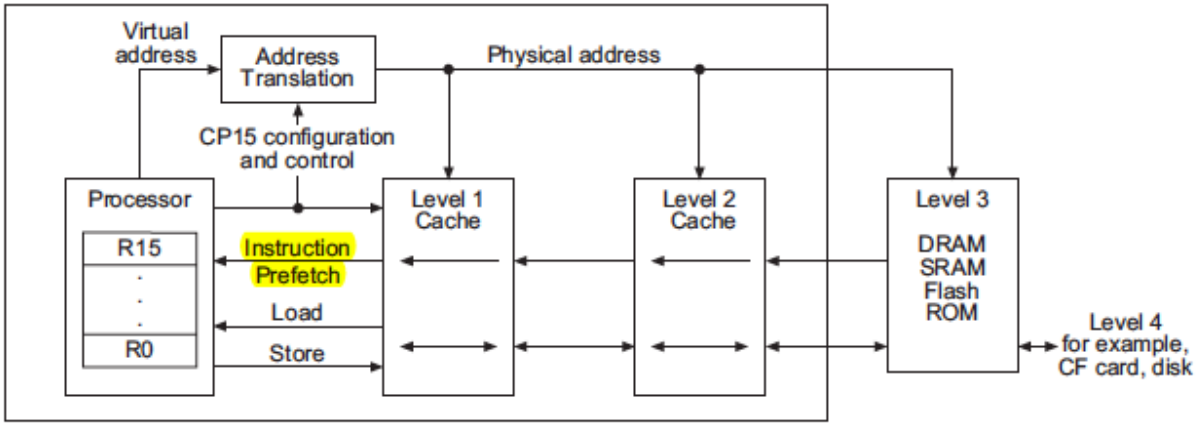


Figure A3-5 Multiple levels of cache in a memory hierarchy

ARMv7 Reference Manual at A3-52 (166) [TPLBN049454], A3-53 – A3-54 (167-68) [TPLBN049455-56]; *see also* ARMv6 Reference Manual at A3-63 (63) [TPLBN035909].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

For implicit accesses:

- Cache linefills and evictions have no effect on the single-copy atomicity of explicit transactions or instruction fetches.
- Instruction fetches are single-copy atomic for each instruction fetched.

Note

32-bit Thumb instructions are fetched as two 16-bit items.

- Translation table walks are performed as 32-bit accesses aligned to 32 bits, each of which is single-copy atomic.

ARMv7 Reference Manual at A3-27 – 3-28 (141-142) [TPLBN049429-30]; *see also* ARMv6 at A3-50 (50) [TPLBN035896].

Table of the AXI bus master IDs

AXI ID	Master bus name	Related IPs
0000_0000	I Block	Camera, JPEG
0000_0001	F Block	Display Controller
0000_0010	P Block	TV Encoder, TV Scaler
XXXX_0011[2]	V Block	MFC
0000_0100	X Block	HSMMC, USB OTG
0000_0101	T Block	Host I/F
0000_0110	M Block	DMA0, DMA1
0000_0111	S Block	Security Sub Block, SDMA0, SDMA1
0000_1000	ARM Instruction	ARM Core Instruction
0000_1001	ARM Data	ARM Core Data
0000_1010	ARM DMA	ARM Core DMA
0000_1011	CF	CFCON
000X_1100[2]	G block	G3D
000X_1101[2]	G block	G3D
0000_1110	G2D	G2D

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

	S3C6410X User Manual at 5-15 (203) [TPLBN002610], 43-8 (1330) [TPLBN003737], *see also* 7-14 (225) [TPLBN002632]. For ARMv6-M, instruction fetches are always halfword-aligned and data accesses are always naturally aligned. Address calculations are normally performed using ordinary integer instructions. This means that they wrap around if they overflow or underflow the address space. Another way of describing this is that any address calculation is reduced modulo 2^{32}. Normal sequential execution of instructions effectively calculates: `(address_of_current_instruction) + (size_of_executed_instruction)` after each instruction to determine the instruction to execute next. If this calculation overflows the top of the address space, the result is UNPREDICTABLE. In ARMv6-M this condition cannot occur because the top of memory is defined to always have the *eXecute Never* (XN) memory attribute associated with it. See *The system address map* on page B3-258 for more information. An access violation is reported if this scenario occurs. The information in this section only applies to instructions that are executed, including those that fail their condition code check. Most ARM implementations prefetch instructions ahead of the currently-executing instruction. ARMv6 Reference Manual at A3-42 (42) [TPLBN035888].
[1L] wherein the means for fetching instructions being configured and connected to fetch multiple sequential instructions from said memory in parallel and supply the multiple	On information and belief, in each Accused Product, the means for fetching instructions are configured and connected to fetch multiple sequential instructions from said memory in parallel and supply the multiple sequential instructions to the central processing unit integrated circuit during a single memory cycle comprises supplying the multiple sequential instructions in parallel to said instruction register during the same memory cycle in which the multiple sequential instructions are fetched. As discussed above in connection with element 1e, the means for fetching instructions is configured and connected to fetch multiple sequential instructions from said memory in parallel and supply the multiple sequential instructions to said central processing unit integrated circuit during a single memory cycle. *See* discussion and evidence above. *See* element 1e, above.

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

sequential instructions to the central processing unit integrated circuit during a single memory cycle comprises supplying the multiple sequential instructions in parallel to said instruction register during the same memory cycle in which the multiple sequential instructions are fetched.	In addition, the single memory cycle during which it is supplied to the CPU is the same memory cycle in which it is fetched. *See, e.g.*: **Instruction Cache Controller** The instruction cache controller fetches the instructions from memory depending on the program flow predicted by the prefetch unit. The instruction cache is 4-way set associative. It comprises the following features: • configurable sizes of 16KB, 32KB, or 64KB • *Virtually Indexed Physically Tagged* (VIPT) • 64-bit native accesses so as to provide up to four instructions per cycle to the prefetch unit • security extensions support • no lockdown support. Cortex-A9 TRM at 7-5 – 7-6 (119-20) [TPLBN034976-77].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

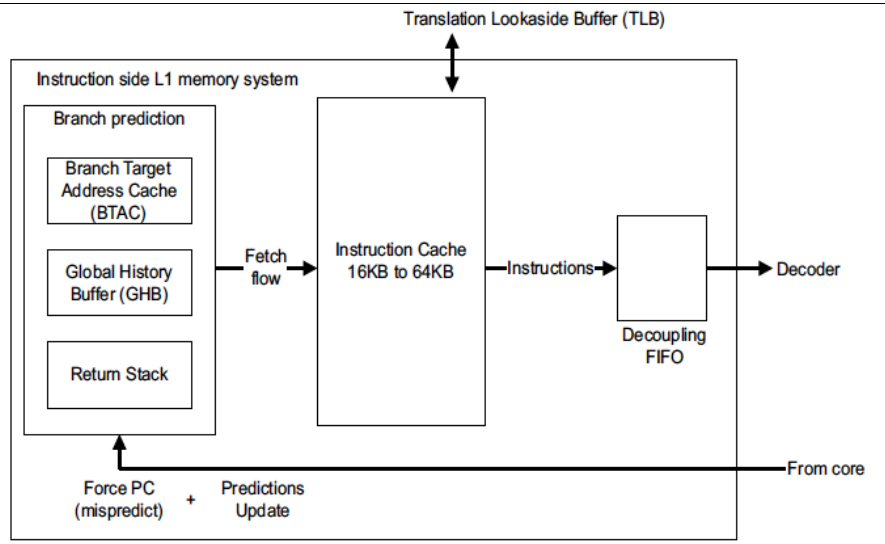


Figure 7-1 Branch prediction and instruction cache

Cortex-A9 TRM at 7-5 (119) [TPLBN034976]

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

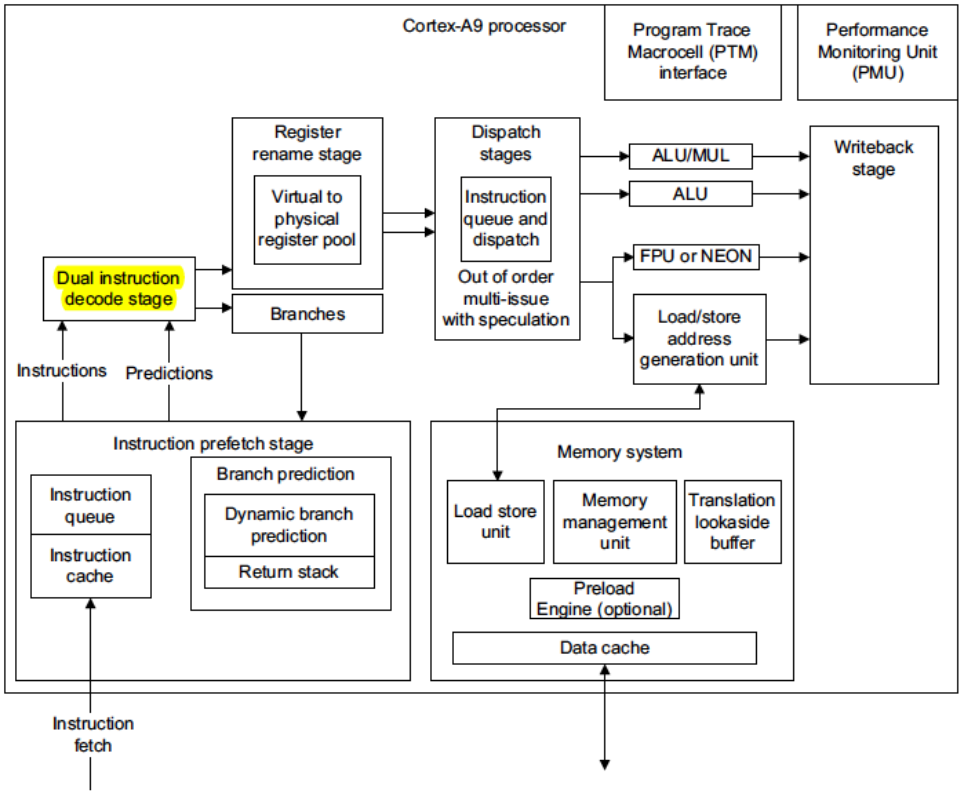


Figure 2-1 Cortex-A9 processor top-level diagram

Cortex-A9 TRM at 2-2 (33) [TPLBN034890].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

Instruction Cache Controller

The instruction cache controller fetches the instructions from memory depending on the program flow predicted by the prefetch unit.

The instruction cache is 4-way set associative. It comprises the following features:

- configurable sizes of 16KB, 32KB, or 64KB
- *Virtually Indexed Physically Tagged* (VIPT)
- 64-bit native accesses so as to provide up to four instructions per cycle to the prefetch unit
- security extensions support
- no lockdown support.

Cortex-A9 TRM at 7-5 – 7-6 (119-20) [TPLBN034976-77].

7.2 Cache organization

Each cache is implemented as a four-way set associative cache of configurable size. The caches are virtually indexed and physically tagged. You can configure the cache sizes in the range of 4 to 64KB. Both the Instruction Cache and the Data Cache can provide two words per cycle for all requesting sources.

ARM11 TRM at 7-3 (374) [TPLBN035461]; Cortex-A8 TRM at 7-3 (244) [TPLBN034521].

Execution stream

The stream of instructions that would have been executed by sequential execution of the program.

ARMv7 Reference Manual at Glossary-5 (2149) [TPLBN051437]; ARMv6 Reference Manual at Glossary-428 (428) [TPLBN036274].

See also Cortex-A8 TRM at 16-13 (504) [TPLBN034781]; Cortex-A9 TRM at B-1 – B-9 (198-207) [TPLBN035055-TPLBN035065]; Cortex-A8 TRM at 2-3 (43) [TPLBN034320].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

Claim 9 (not asserted)[3]	
[9 preamble] A microprocessor system, comprising	On information and belief, each Accused Product listed in the attached list of Accused Products contains a microprocessor ("Accused Microprocessors"). Each microprocessor is an electronic circuit that interprets and executes programmed instructions. *See* claim chart for claim 1, above.
[9a] a central processing unit,	On information and belief, each Accused Product contains one of the Accused Microprocessors, each of which contains one or more central processing units. *See* element 1a, above.
[9b] a dynamic random access memory,	On information and belief, each Accused Product has a dynamic random access memory. *See* element 1b, above.
[9c] a bus connecting said central processing unit to said dynamic random access memory, and	On information and belief, each Accused Product has a bus connecting said central processing unit to said dynamic random access memory. *See* element 1c, above.
[9d] multiplexing means on said bus	On information and belief, each Accused Product has multiplexing means on said bus between said central processing unit and said dynamic random access memory.

[3] Claim 9 is not asserted, but is included in this chart because it is the independent claim from which dependent claim 59 depends.

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

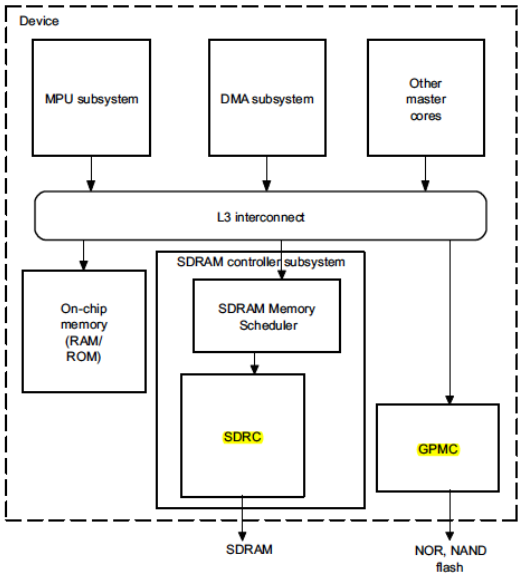
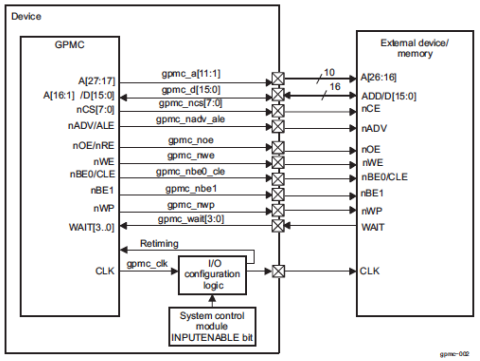
between said central processing unit and said dynamic random access memory,	*See* elements 1c and 1d, above. **Figure 10-1. GPMC Environment**  • SDRAM Controller – Support for two independent CSs, with their corresponding register sets, and independent page tracking – Supports the following memory types: • Mobile Single Data Rate SDRAM (M-SDR) • Low-Power Double Data Rate SDRAM (LPDDR) – Memory device capacity: • 16 Mbits, 32 Mbits, 64 Mbits, 128 Mbits, 256 Mbits, 512 Mbits , 1 Gbit, and 2 Gbits device support – Memory device organization • 2-bank support for 16 Mbits and 32 Mbits • 4-bank support for 64 Mbits to 2 Gbits • Flexible row/column address multiplexing schemes • Bank linear addressing • 16- or 32-bit data path to external SDRAM memory • 1GB maximum addressing capability (256MB per chip-select) • Device driver strength feature for mobile DDR supported • New flexible address-muxing scheme lets users choose different bank mapping allocations by configuring the bank and column address decoding ordering. OMAP36xx TRM at 2240 [TPLBN006952], 2241 [TPLBN006953]. **Figure 10-2. GPMC to 16-Bit Address/Data-Multiplexed Memory**  **10.1.1.1 GPMC Features** The GPMC is the device 16-bit external memory controller. The GPMC data access engine provides a flexible programming model for communication with all standard memories. The GPMC supports various accesses: • Asynchronous read/write access • Asynchronous read page access (4, 8, 16 Word16) • Synchronous read/write access • Synchronous read/write burst access without wrap capability (4, 8, 16 Word16) • Synchronous read/write burst access with wrap capability (4, 8, 16 Word16) • Address/data-multiplexed access • Little- and big-endian access OMAP36xx TRM at 2229-30 [TPLBN006842-43]; *see also* 2132 [TPLBN006845].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

10.1.3.3 GPMC Address and Data Bus

The current application supports GPMC connection to address/data-multiplexed memory and a NAND device. Connection to a nonmultiplexed address/data memory is supported with an address range of only 2 Kbytes.

Depending on the GPMC configuration on each chip-select, address and data-bus lines that are not required for a particular access protocol are not updated (changed from current value) and are not sampled when input (input data bus).

The current application supports GPMC connection to address/data-multiplexed memory, address/data-nonmultiplexed memory with limited address (2 Kbytes), and a NAND device:

- When the GPMC.GPMC_CONFIG[1] LIMITEDADDRESS bit is set to 1, only gpmc_a[11:1] address lines are used. This limits the memory support to 2K-byte addressable memories.
- For address/data-multiplexed NOR devices, the address is multiplexed on the data bus.
- 8-bit wide NOR devices do not use GPMC I/O: gpmc_d[15:8] for data (they are used for address if needed).
- 16-bit wide NAND devices do not use GPMC I/O: gpmc_a[11:1] .
- 8-bit wide NAND devices do not use GPMC I/O: gpmc_a[11:1] and GPMC I/O: gpmc_d[15:8].

OMAP36xx TRM at 2135 [TPLBN006848].

10.2.2.2.3 Address Multiplexing

A flexible address scheme has been added to support any new type of address scheme and SDRAM density. This new address-muxing scheme lets users choose a different bank mapping allocation by configuring the order of the bank and row address decoding (column address always remains the same). The SDRC.SDRC_MCFG_p[7:6] BANKALLOCATION field defines the order of the bank and row decoding of the incoming system address. The BANKALLOCATION field can be set on a CS basis. For more details about the flexible address scheme, see Section 10.2.4.4.3, *Address Multiplexing*.

The programming model is compatible with the legacy fixed address scheme and with the new flexible address scheme. This section describes the fixed address multiplexing configurations for both SDRAM components.

The legacy fixed address scheme is selected when the SDRC.SDRC_MCFG_p[19] ADDRMUXLEGACY bit is set to 0 (where p = 0 or 1 for SDRC CS0 or CS1). The fixed address multiplexing configurations are described in this section, for both SDRAM components. An address multiplexing scheme is chosen by programming the SDRC.SDRC_MCFG_p[24:20] ADDRMUX field of the SDRC_MEMCFG registers on a per chip select basis (p = 0 or 1 for CS0 or CS1). Table 10-96 and Table 10-97 show all predefined address multiplexing schemes for SDRAM memory components.

OMAP36xx TRM at 2244 [TPLBN006957]; *see also* 2265 [TPLBN006978], 2245 [TPLBN006958]

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

10.1.5.2.3 Address/Data-Multiplexing Interface

For random synchronous or asynchronous memory interfacing (DEVICETYPE = 0b00), an address- and data-multiplexing protocol can be selected through the GPMC.GPMC_CONFIG1_i[9] MUXADDDATA bit (i = 0 to 7). The nADV signal must be used as the external device address latch control signal. For the associated chip-select configuration, nADV assertion and deassertion time and nOE assertion time must be set to the appropriate value to meet the address latch setup/hold time requirements of the external device. See Section 10.1.3, *GPMC Integration*.

OMAP36xx TRM at 2241 [TPLBN006854].

Figure 16-1. Memory Subsystem Functional Diagram

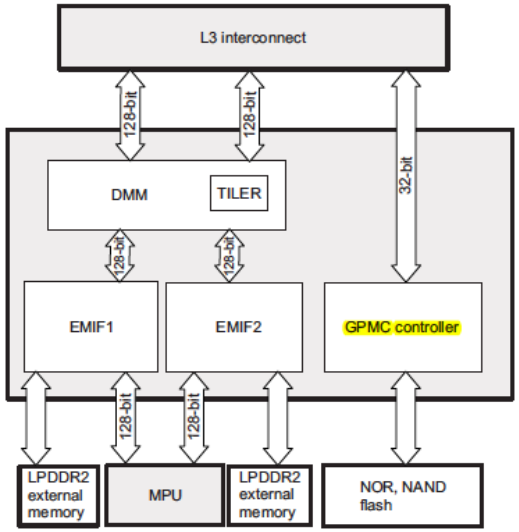


Figure 16-50. GPMC to 16-Bit Address/Data-Multiplexed Memory

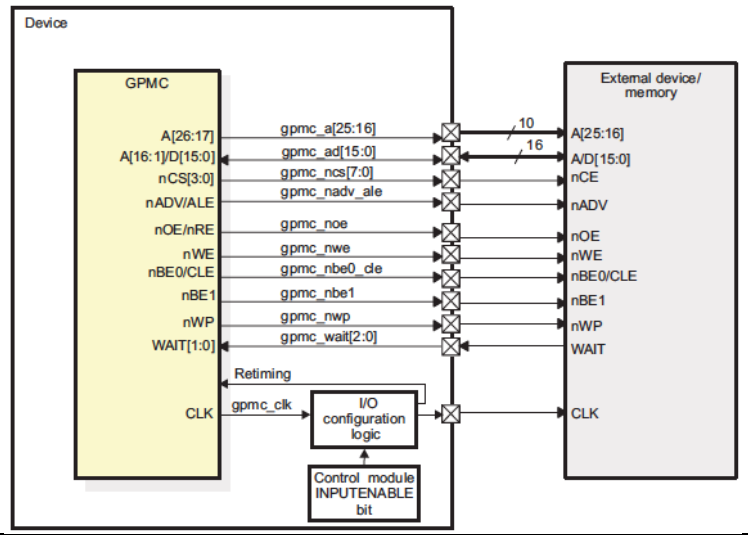


EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

16.1.4.1 GPMC Features

The GPMC is the external memory controller of the device. The GPMC data access engine provides a flexible programming model for communication with all standard memories. The GPMC supports various accesses:

- Asynchronous read/write access
- Asynchronous read page access (4, 8, and 16 Word16)
- Synchronous read/write access
- Synchronous read/write burst access without wrap capability (4, 8, and 16 Word16)
- Synchronous read/write burst access with wrap capability (4, 8, and 16 Word16)
- Address/data-multiplexed access
- Little- and big-endian access

OMAP4470 TRM at 3280 [TPLBN024480], 3282-83 [TPLBN024482-83], 3438 [TPLBN024638]; *see also* 3440 [TPLBN024640]; 3437 [TPLBN024637].

16.4.4 GPMC Functional Description

The GPMC basic programming model offers maximum flexibility to support various access protocols for each of the eight configurable chip-selects. Use optimal chip-select settings, based on the characteristics of the external device:

- Different protocols can be selected to support generic asynchronous or synchronous random-access devices (NOR flash, SRAM) or to support specific NAND devices.
- The address and the data bus can be multiplexed on the same external bus.
- Read and write access can be independently defined as asynchronous or synchronous.
- System requests (byte, 16-bit word, burst) are performed through single or multiple accesses. External access profiles (single, multiple with optimized burst length, native- or emulated-wrap) are based on external device characteristics (supported protocol, bus width, data buffer size, native-wrap support).
- System burst read or write requests are synchronous-burst (multiple-read or multiple-write). When neither burst nor page mode is supported by external memory or ASIC devices, system burst read or write requests are translated to successive single synchronous or asynchronous accesses (single reads or single writes). 8-bit wide devices are supported only in single synchronous or single asynchronous read or write mode.
- To simulate a programmable internal-wait state, an external wait pin can be monitored to dynamically control external access at the beginning (initial access time) of and during a burst access.

OMAP4470 TRM at 3444 [TPLBN024644].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

	16.4.4.8 GPMC Address and Data Bus The current application supports GPMC connection to NAND devices and to address/data-multiplexed memories or devices. Connection to address/data-nonmultiplexed memories or devices is supported with a limited address range of 2 Kbytes. Depending on the GPMC configuration of each chip-select, address and data bus lines that are not required for a particular access protocol are not updated (changed from current value) and are not sampled when input (input data bus). • When the GPMC_CONFIG[1] LIMITEDADDRESS bit is set to 1, A26-A11 are not modified during an external memory access. This limits the memory address space to 2K-byte regardless of the memory size. • For address/data-multiplexed and AAD-multiplexed NOR devices, the address is multiplexed on the data bus. • 8-bit wide NOR devices do not use GPMC I/O: gpmc_ad[15:8] for data (they are used for address if needed). • 16-bit wide NAND devices do not use GPMC I/O: gpmc_a[25:16]. • 8-bit wide NAND devices do not use GPMC I/O: gpmc_a[25:16] and GPMC I/O: gpmc_ad[15:8]. OMAP4470 TRM at 3448 [TPLBN024648]. ***16.4.4.9.2.3 Address/Data-Multiplexing Interface*** For random synchronous or asynchronous memory interfacing (DEVICETYPE = 0b00), an address- and data-multiplexing protocol can be selected through the GPMC_CONFIG1_i[9:8] MUXADDDATA bit field (i = 0 to 7). The nADV signal must be used as the external device address latch control signal. For the associated chip-select configuration, nADV assertion and deassertion time and nOE assertion time must be set to the appropriate value to meet the address latch setup/hold time requirements of the external device. See Section 16.4.3, *GPMC Integration*. OMAP4470 TRM at 3452 [TPLBN024652].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

Figure 15-1. Memory Subsystem Functional Diagram

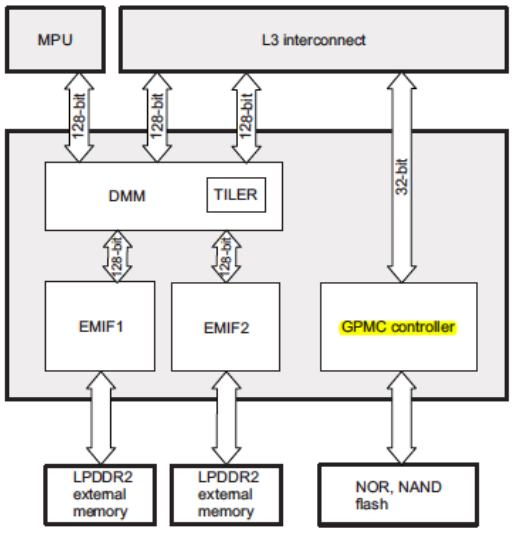
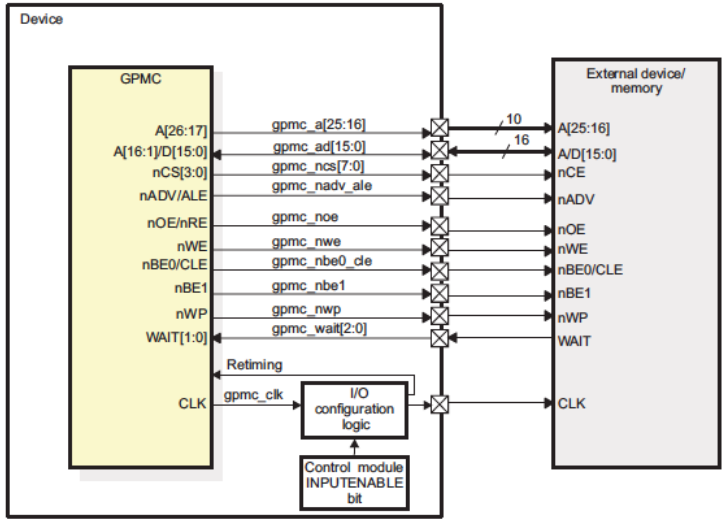


Figure 15-50. GPMC to 16-Bit Address/Data-Multiplexed Memory



15.1.4.1 GPMC Features

The GPMC is the external memory controller of the device. The GPMC data access engine provides a flexible programming model for communication with all standard memories. The GPMC supports various accesses:

- Asynchronous read/write access
- Asynchronous read page access (4, 8, and 16 Word16)
- Synchronous read/write access
- Synchronous read/write burst access without wrap capability (4, 8, and 16 Word16)
- Synchronous read/write burst access with wrap capability (4, 8, and 16 Word16)
- Address/data-multiplexed access
- Little- and big-endian access

OMAP4430 TRM at 3207 [TPLBN011706], 3209-10 [TPLBN011708-10], 3366 [TPLBN011865]; *see also* 3368 [TPLBN011865]; 3376 [TPLBN011875].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

	15.4.4 GPMC Functional Description The GPMC basic programming model offers maximum flexibility to support various access protocols for each of the eight configurable chip-selects. Use optimal chip-select settings, based on the characteristics of the external device: • Different protocols can be selected to support generic asynchronous or synchronous random-access devices (NOR flash, SRAM) or to support specific NAND devices. • The address and the data bus can be multiplexed on the same external bus. • Read and write access can be independently defined as asynchronous or synchronous. • System requests (byte, 16-bit word, burst) are performed through single or multiple accesses. External access profiles (single, multiple with optimized burst length, native- or emulated-wrap) are based on external device characteristics (supported protocol, bus width, data buffer size, native-wrap support). • System burst read or write requests are synchronous-burst (multiple-read or multiple-write). When neither burst nor page mode is supported by external memory or ASIC devices, system burst read or write requests are translated to successive single synchronous or asynchronous accesses (single reads or single writes). 8-bit wide devices are supported only in single synchronous or single asynchronous read or write mode. • To simulate a programmable internal-wait state, an external wait pin can be monitored to dynamically control external access at the beginning (initial access time) of and during a burst access. OMAP4430 TRM at 3372 [TPLBN011871]. **15.4.4.8 GPMC Address and Data Bus** The current application supports GPMC connection to NAND devices and to address/data-multiplexed memories or devices. Connection to address/data-nonmultiplexed memories or devices is supported with a limited address range of 2 Kbytes. Depending on the GPMC configuration of each chip-select, address and data bus lines that are not required for a particular access protocol are not updated (changed from current value) and are not sampled when input (input data bus). • When the GPMC_CONFIG[1] LIMITEDADDRESS bit is set to 1, A26-A11 are not modified during an external memory access. This limits the memory address space to 2K-byte regardless of the memory size. • For address/data-multiplexed and AAD-multiplexed NOR devices, the address is multiplexed on the data bus. • 8-bit wide NOR devices do not use GPMC I/O: gpmc_ad[15:8] for data (they are used for address if needed). • 16-bit wide NAND devices do not use GPMC I/O: gpmc_a[25:16]. • 8-bit wide NAND devices do not use GPMC I/O: gpmc_a[25:16] and GPMC I/O: gpmc_ad[15:8]. OMAP4430 TRM at 3376 [TPLBN011875].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

	15.4.4.9.2.3 Address/Data-Multiplexing Interface For random synchronous or asynchronous memory interfacing (DEVICETYPE = 0b00), an address- and data-multiplexing protocol can be selected through the GPMC_CONFIG1_i[9:8] MUXADDDATA bit field (i = 0 to 7). The nADV signal must be used as the external device address latch control signal. For the associated chip-select configuration, nADV assertion and deassertion time and nOE assertion time must be set to the appropriate value to meet the address latch setup/hold time requirements of the external device. See Section 15.4.3, *GPMC Integration*. OMAP4430 TRM at 3380 [TPLBN011879]. **15.3.1.1 EMIF Module Main Features** The EMIF module supports the following features: • JEDEC standard compliant LPDDR2-SDRAM (S2 and S4) and LPDDR2-NVM devices. • Two GB SDRAM address range over two chip-selects (1GB per CS) (configurable with the DMM module, see Section 15.2, *Dynamic Memory Manager* for more information) • Two independent chip-selects with their corresponding register sets and independent page tracking • Both chip-selects must have the same memory type and size if they are both SDRAM or both NVM. LPDDR2-SDRAM can be used in parallel with LPDDR2-NVM and can have a different size. • Flexible address muxing scheme which permit to choose different bank mapping allocation by configuring the bank, column and row address decoding ordering • 16 or -32-bit data path to external SDRAM memory • LPDDR2 devices with 1, 2. 4 or 8 of internal banks • Data bus widths: – 128-bit L3 interconnect Data Bus Width – 16 and 32-bit SDRAM Data Bus Width • CAS latencies: 3, 4, 5, 6, 7 and 8 • 256, 512, 1024, and 2048-word page sizes • Burst lenghts: 8 • Sequential burst type • SDRAM auto initialization from reset or configuration change • Bank Interleaving across both the chip-selects if same memory type OMAP4430 TRM at 3292 [TPLBN011791].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

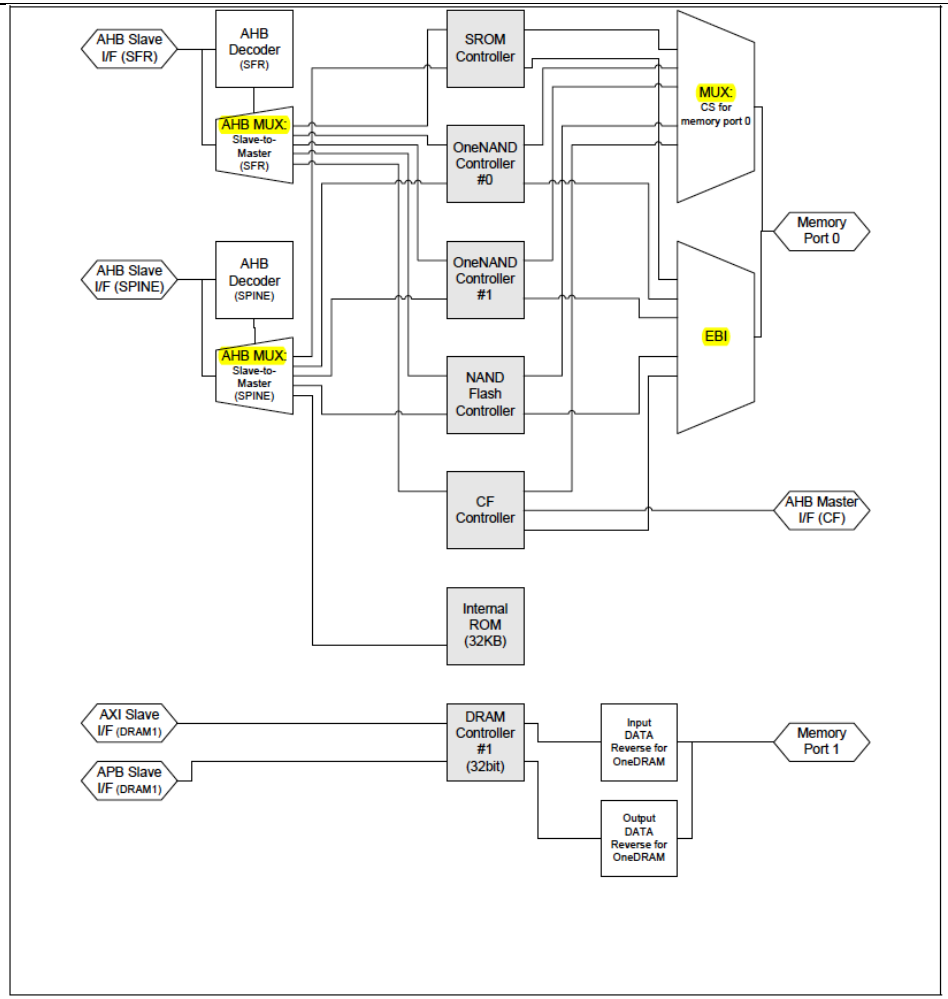


Figure 4-1 Structure of Memory Sub-system

S3C6410X User Manual at 4-4 (176) [TPLBN002583].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

4.3. STRUCTURE

Figure 4-1 shows the structure of Memory sub-system. It holds eight memory controllers, which are shown in grey boxes. Interface ports of Memory sub-system includes two AHB slave ports, each of which is for accessing SFRs and memory contents, two APB slave ports , two AXI slave ports, single AHB master ports, and two memory interface ports. Configuration signal ports, which changes the operation options of Memory Sub-system, are not shown in Figure 4-1 for clarity. Description of these configuration signal ports are detailed in later section. For memory port 0, EBI is used to multiplex output signals coming from each memory controller to generate a unified control signals for memory devices. Figure 4-1 also shows "reverse data" block, which reverses the order of bit assignments of data port of DRAM controller #1. This is useful when OneDRAM is attached to DRAM controller #1 as OneDRAM has reverse data port order compared to mobile DRAM.

S3C6410X User Manual at 4-1 (174) [TPLBN002581].

4.6. EBI MULTIPLEXING

EBI is used to multiplex output signals coming from each memory controller to generate a unified control signals for memory devices. Table 4-3 shows which of memory controller participates in to generate each memory control signals.

S3C6410X User Manual at 4-1 (177) [TPLBN002584], *see also* 7-3 (214) [TPLBN002621].

9.2.1 THE CF CONTROLLER FEATURES:

The CF controller supports only 1 slot.

The CF controller consists of 2 parts – PC card controller & ATA controller. They are multiplexing from or to PAD signals. You must use only 1 mode, PC card or True-IDE mode. Default mode is PC card mode. The CF controller has a top level SFR that includes card power enable bit, output port enable bit & mode select (True-IDE or PC card) bit.

S3C6410X User Manual at 9-1 (274) [TPLBN002681].

On information and belief, the DDR memory components shown in the Accused Products require a multiplexing means connected to the bus between the memory and the CPU. *See, e.g.*, http://www.jedec.org; *see also* JESD209-2E, available online at http://www.jedec.org/sites/default/files/docs/JESD209-2E.pdf at 3; JEDEC Standard No. 21-C, available online at http://www.jedec.org/sites/default/files/2_00R20.pdf.

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

[9e] said multiplexing means being connected and configured to provide multiplexed row addresses, column addresses and data on said bus from said central processing unit to said dynamic random access memory and to provide data from said dynamic random access memory to said central processing unit, and	On information and belief, each Accused Product has multiplexing means connected and configured to provide multiplexed row addresses, column addresses and data on said bus from said central processing unit to said dynamic random access memory and to provide data from said dynamic random access memory to said central processing unit. *See* element 9d, above. The multiplexing means multiplexes row addresses, column addresses, and data on the bus from the CPU to the DRAM and to provide the data from the DRAM. The controllers described in connection with element 9d above are all connected to the bus system for the various microprocessor chips in the Accused Products, and are all capable of passing multiplexed information between the CPU and memory. As shown in the documents cited above in connection with element 9d, the multiplexed information includes multiplexing of row/column addresses and data. *See, for example:*

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

	• SDRAM Controller – Support for two independent CSs, with their corresponding register sets, and independent page tracking – Supports the following memory types: • Mobile Single Data Rate SDRAM (M-SDR) • Low-Power Double Data Rate SDRAM (LPDDR) – Memory device capacity: • 16 Mbits, 32 Mbits, 64 Mbits, 128 Mbits, 256 Mbits, 512 Mbits , 1 Gbit, and 2 Gbits device support – Memory device organization • 2-bank support for 16 Mbits and 32 Mbits • 4-bank support for 64 Mbits to 2 Gbits • Flexible row/column address multiplexing schemes • Bank linear addressing • 16- or 32-bit data path to external SDRAM memory • 1GB maximum addressing capability (256MB per chip-select) • Device driver strength feature for mobile DDR supported • New flexible address-muxing scheme lets users choose different bank mapping allocations by configuring the bank and column address decoding ordering. OMAP36xx TRM at 2241 [TPLBN006953], *see also* 2240 [TPLBN006952].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

	10.1.3.3 GPMC Address and Data Bus The current application supports GPMC connection to address/data-multiplexed memory and a NAND device. Connection to a nonmultiplexed address/data memory is supported with an address range of only 2 Kbytes. Depending on the GPMC configuration on each chip-select, address and data-bus lines that are not required for a particular access protocol are not updated (changed from current value) and are not sampled when input (input data bus). The current application supports GPMC connection to address/data-multiplexed memory, address/data-nonmultiplexed memory with limited address (2 Kbytes), and a NAND device: • When the GPMC.GPMC_CONFIG[1] LIMITEDADDRESS bit is set to 1, only gpmc_a[11:1] address lines are used. This limits the memory support to 2K-byte addressable memories. • For address/data-multiplexed NOR devices, the address is multiplexed on the data bus. • 8-bit wide NOR devices do not use GPMC I/O: gpmc_d[15:8] for data (they are used for address if needed). • 16-bit wide NAND devices do not use GPMC I/O: gpmc_a[11:1] . • 8-bit wide NAND devices do not use GPMC I/O: gpmc_a[11:1] and GPMC I/O: gpmc_d[15:8]. OMAP36xx TRM at 2135 [TPLBN006848]; *see also* OMAP4470 TRM at 3448 [TPLBN024648]; OMAP4430 TRM at 3376 [TPLBN011875]. ***10.1.5.2.3 Address/Data-Multiplexing Interface*** For random synchronous or asynchronous memory interfacing (DEVICETYPE = 0b00), an address- and data-multiplexing protocol can be selected through the GPMC.GPMC_CONFIG1_i[9] MUXADDDATA bit (i = 0 to 7). The nADV signal must be used as the external device address latch control signal. For the associated chip-select configuration, nADV assertion and deassertion time and nOE assertion time must be set to the appropriate value to meet the address latch setup/hold time requirements of the external device. See Section 10.1.3, *GPMC Integration*. OMAP36xx TRM at 2241 [TPLBN006854]; *see also* OMAP4470 TRM at 3452 [TPLBN024652]; OMAP4430 TRM at 3380 [TPLBN011879].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

3.2.55 c9, L2 Cache Auxiliary Control Register

The purpose of the L2 Cache Auxiliary Control Register is to enable you to configure the L2 cache behavior.

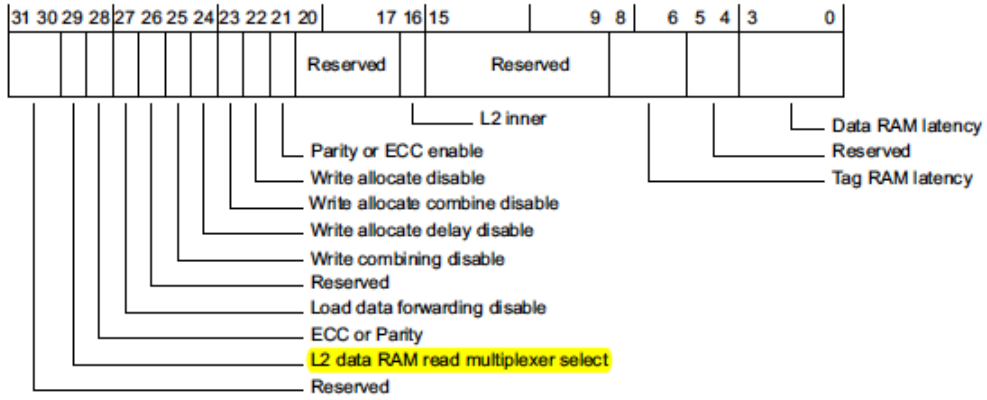
The L2 Cache Auxiliary Control Register is:

- a read register, common to Secure and Nonsecure states
- a write register in Secure state only
- accessible in privileged modes only.

Note

If bit [24] of the L2 Cache Auxiliary Control Register is not set to 1, the L2 cache does not perform a security check for the data that is placed in the L2 cache. If software requires a higher level of security within the processor, then you must set bit [24] to 1 of this register. By setting bit [24] to 1, the L2 cache performs an external linefill, and the AXI slave performs the security check on that linefill.

Figure 3-49 shows the bit arrangement of the L2 Cache Auxiliary Control Register.



Cortex A8 TRM at 3-95 (173) [TPLBN034450].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

Bits	Field	Function
[31:30]	-	Reserved. UNP, SBZP.
[29]	L2 data RAM read multiplexer select	Configures the timing of the read data multiplexer select between one or two cycles for all L2 data RAM read operations: 0 = two cycles, default 1 = one cycle.

Cortex A8 TRM at 3-95 (173) [TPLBN034450].

16.4.4 GPMC Functional Description

The GPMC basic programming model offers maximum flexibility to support various access protocols for each of the eight configurable chip-selects. Use optimal chip-select settings, based on the characteristics of the external device:

- Different protocols can be selected to support generic asynchronous or synchronous random-access devices (NOR flash, SRAM) or to support specific NAND devices.
- The address and the data bus can be multiplexed on the same external bus.
- Read and write access can be independently defined as asynchronous or synchronous.
- System requests (byte, 16-bit word, burst) are performed through single or multiple accesses. External access profiles (single, multiple with optimized burst length, native- or emulated-wrap) are based on external device characteristics (supported protocol, bus width, data buffer size, native-wrap support).
- System burst read or write requests are synchronous-burst (multiple-read or multiple-write). When neither burst nor page mode is supported by external memory or ASIC devices, system burst read or write requests are translated to successive single synchronous or asynchronous accesses (single reads or single writes). 8-bit wide devices are supported only in single synchronous or single asynchronous read or write mode.
- To simulate a programmable internal-wait state, an external wait pin can be monitored to dynamically control external access at the beginning (initial access time) of and during a burst access.

OMAP4470 TRM at 3444 [TPLBN024644]; *see also* OMAP4430 TRM at 3372 [TPLBN011871].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

4.3. STRUCTURE

Figure 4-1 shows the structure of Memory sub-system. It holds eight memory controllers, which are shown in grey boxes. Interface ports of Memory sub-system includes two AHB slave ports, each of which is for accessing SFRs and memory contents, two APB slave ports , two AXI slave ports, single AHB master ports, and two memory interface ports. Configuration signal ports, which changes the operation options of Memory Sub-system, are not shown in Figure 4-1 for clarity. Description of these configuration signal ports are detailed in later section. For memory port 0, EBI is used to multiplex output signals coming from each memory controller to generate a unified control signals for memory devices. Figure 4-1 also shows "reverse data" block, which reverses the order of bit assignments of data port of DRAM controller #1. This is useful when OneDRAM is attached to DRAM controller #1 as OneDRAM has reverse data port order compared to mobile DRAM.

S3C6410X User Manual at 4-1 (174) [TPLBN002581]; *see also* 4-5 (177) [TPLBN002584], 7-3 (214) [TPLBN002621], 9-1 (274) [TPLBN002681].

15.4.4.8 GPMC Address and Data Bus

The current application supports GPMC connection to NAND devices and to address/data-multiplexed memories or devices. Connection to address/data-nonmultiplexed memories or devices is supported with a limited address range of 2 Kbytes.

Depending on the GPMC configuration of each chip-select, address and data bus lines that are not required for a particular access protocol are not updated (changed from current value) and are not sampled when input (input data bus).

- When the GPMC_CONFIG[1] LIMITEDADDRESS bit is set to 1, A26-A11 are not modified during an external memory access. This limits the memory address space to 2K-byte regardless of the memory size.
- For address/data-multiplexed and AAD-multiplexed NOR devices, the address is multiplexed on the data bus.
- 8-bit wide NOR devices do not use GPMC I/O: gpmc_ad[15:8] for data (they are used for address if needed).
- 16-bit wide NAND devices do not use GPMC I/O: gpmc_a[25:16].
- 8-bit wide NAND devices do not use GPMC I/O: gpmc_a[25:16] and GPMC I/O: gpmc_ad[15:8].

OMAP4430 TRM at 3376 [TPLBN011875].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

	15.4.4 GPMC Functional Description The GPMC basic programming model offers maximum flexibility to support various access protocols for each of the eight configurable chip-selects. Use optimal chip-select settings, based on the characteristics of the external device: • Different protocols can be selected to support generic asynchronous or synchronous random-access devices (NOR flash, SRAM) or to support specific NAND devices. • The address and the data bus can be multiplexed on the same external bus. • Read and write access can be independently defined as asynchronous or synchronous. • System requests (byte, 16-bit word, burst) are performed through single or multiple accesses. External access profiles (single, multiple with optimized burst length, native- or emulated-wrap) are based on external device characteristics (supported protocol, bus width, data buffer size, native-wrap support). • System burst read or write requests are synchronous-burst (multiple-read or multiple-write). When neither burst nor page mode is supported by external memory or ASIC devices, system burst read or write requests are translated to successive single synchronous or asynchronous accesses (single reads or single writes). 8-bit wide devices are supported only in single synchronous or single asynchronous read or write mode. • To simulate a programmable internal-wait state, an external wait pin can be monitored to dynamically control external access at the beginning (initial access time) of and during a burst access. OMAP4430 TRM at 3372 [TPLBN011871]. In addition, on information and belief, the additional discovery will show that each of the memory devices in the accused products sends and receives information that is multiplexed to and from the CPU by way of a multiplexing means connected to the bus that multiplexes row/column addresses and data.
[9f] means connected to said bus for fetching instructions for said central processing unit on said bus from said dynamic random access memory,	On information and belief, each Accused Product has multiplexing means connected to said bus for fetching instructions for said central processing unit on said bus from said dynamic random access memory. *See* element 1d, above.

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

[9g] said means for fetching instructions being configured to fetch multiple sequential instructions from said dynamic random access memory in parallel and supply the multiple instructions to said central processing unit during a single memory cycle,	On information and belief, in each Accused Product, said means for fetching instructions is configured to fetch multiple sequential instructions from said dynamic random access memory in parallel and supply the multiple instructions to said central processing unit during a single memory cycle. *See* element 1e, above.
[9h] said central processing unit including an arithmetic logic unit and a first push down stack connected to said arithmetic logic unit,	On information and belief, in each Accused Product, said central processing unit includes an arithmetic logic unit and a first push down stack connected to said arithmetic logic unit. *See* element 1g, above.
[9i] said first push down stack	On information and belief, each Accused Product has a first push down stack including means for storing a top item connected to a first input of said arithmetic logic unit to provide the top item to the first input and means for storing a next item connected to a second input of said arithmetic logic unit to provide the next item to the second input.

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

including means for storing a top item connected to a first input of said arithmetic logic unit to provide the top item to the first input, and means for storing a next item connected to a second input of said arithmetic logic unit to provide the next item to the second input,	*See* elements 1h and 1i, above.
[9j] a remainder of said first push down stack being connected to said means for storing a next item to receive the next item from said means for storing a next item when pushed down in said push down stack,	On information and belief, in each Accused Product, a remainder of said first push down stack is connected to said means for storing a next item to receive the next item from said means for storing a next item when pushed down in said push down stack. *See* elements 1h and 1i, above.
[9k]	On information and belief, each Accused Product has an arithmetic logic unit having an output connected to said means for storing a top item.

said arithmetic logic unit having an output connected to said means for storing a top item.	*See* elements 1g, 1h, 1i, and 1j, above.

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

Claim 43	
[43 preamble] The microprocessor system of claim 1	As shown in connection with claim 1, on information and belief, each Accused Product contains one of the Accused Microprocessors and includes all elements of claim 1. *See* claim chart for claim 1, above.
[43a] wherein said central processing unit integrated circuit a program counter comprising address bits,	On information and belief, each Accused Product includes a central processing unit integrated circuit a program counter comprising address bits. As discussed above in connection with claim 1, on information and belief, each Accused Product includes a central processing unit integrated circuit. *See* element 1a, above. The Accused Products include a program counter (PC): ***7.1.2 Features*** The dual Cortex-M3 MPU subsystem integrates the following: • Two ARM Cortex-M3 microprocessors: – ARMv7-M and Thumb®-2 instruction set architecture – Hardware division and single-cycle multiplication computational acceleration – Integrated nested vector interrupt controller (NVIC) – Integrated bus matrix – Registers: • Thirteen general-purpose 32-bit registers • Link register (LR) • Program counter (PC) • Program status register, xPSR • Two banked SP registers – Integrated power management – Extensive debug capabilities OMAP4470 TRM at 1495 [TPLBN022695]; *see also* OMAP4430 at 1457 [TPLBN009956].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

2.13.1 The state register set

In ARM state, 16 data registers and one or two status registers are accessible at any time. In privileged modes, mode-specific banked registers become available. Figure 2-10 on page 2-19 shows the registers that are available in each mode.

Thumb and ThumbEE state give access to the same set of registers as ARM state. However, the 16-bit instructions provide only limited access to some of the registers. No such limitations exist for 32-bit Thumb-2 and ThumbEE instructions.

Registers r0 through r13 are general-purpose registers used to hold either data or address values.

Registers r14 and r15 have the following special functions:

Link Register Register r14 is used as the subroutine *Link Register* (LR).

Register r14 receives the return address when the processor executes a *Branch with Link* (BL or BLX) instruction.

You can treat r14 as a general-purpose register at all other times. Similarly, the corresponding banked registers r14_mon, r14_svc, r14_irq, r14_fiq, r14_abt, and r14_und hold the return values when the processor receives interrupts and exceptions, or when it executes the BL or BLX instructions within interrupt or exception routines.

Program Counter Register r15 holds the PC:

- in ARM state, this is word-aligned
- in Thumb state, this is halfword-aligned
- in ThumbEE state, this is halfword-aligned.

Cortex-A8 TRM at 58 [TPLBN034335]; *see also* 59-60 [TPLBN034336-37]; Cortex-A9 TRM at 34 [TPLBN034891]; ARM11 TRM at 91 [TPLBN035177], 93-94 [TPLBN035180-81]; ARM Architecture Reference Manual at A1-3 (25) [PIC00005215], A1-6 (28) [PIC00005218], A2-4 (36) [PIC00005226].

The program counter in the Accused Products stores address bits.

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

2.15.1 Exception entry and exit summary

Table 2-12 summarizes the PC value preserved in the relevant r14 on exception entry and the recommended instruction for exiting the exception handler.

Table 2-12 Exception entry and exit

Exception or entry	Return instruction	Previous state		Notes
		ARM r14_x	Thumb r14_x	
SVC	MOVS PC, R14_svc	PC + 4	PC+2	Where the PC is the address of the SVC, SMC, or Undefined instruction
SMC	MOVS PC, R14_mon	PC + 4	-	
UNDEF	MOVS PC, R14_und	PC + 4	PC+2	
PABT	SUBS PC, R14_abt, #4	PC + 4	PC+4	Where the PC is the address of instruction that had the prefetch abort
FIQ	SUBS PC, R14_fiq, #4	PC + 4	PC+4	Where the PC is the address of the instruction that was not executed because the FIQ or IRQ took priority
IRQ	SUBS PC, R14_irq, #4	PC + 4	PC+4	
DABT	SUBS PC, R14_abt, #8	PC + 8	PC+8	Where the PC is the address of the load or store instruction that generated the data abort
RESET	-	-	-	The value saved in r14_svc on reset is Unpredictable
BKPT	SUBS PC, R14_abt, #4	PC + 4	PC+4	Software breakpoint

Cortex-A8 TRM at 67 [TPLBN034344]; *see also* Cortex-A9 TRM at 34 [TPLBN034891], 58-62 [TPLBN034915-19] (Table 4-1); ARM11 TRM at 110 [TPLBN035197]; ARM Architecture Reference Manual at A2-7 – A2-8 (39-40) [PIC00005229-30].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

LR, the Link Register

Register R14 is used to store the return address from a subroutine. At other times, LR can be used for other purposes.

When a BL or BLX instruction performs a subroutine call, LR is set to the subroutine return address. To perform a subroutine return, copy LR back to the program counter. This is typically done in one of two ways, after entering the subroutine with a BL or BLX instruction:

- Return with a `BX LR` instruction.
- On subroutine entry, store LR to the stack with an instruction of the form:
  `PUSH {<registers>,LR}`
  and use a matching instruction to return:
  `POP {<registers>,PC}`

ThumbEE checks and handler calls use LR in a similar way. For details see Chapter A9 *ThumbEE*.

PC, the Program Counter

Register R15 is the program counter:

- When executing an ARM instruction, PC reads as the address of the current instruction plus 8.
- When executing a Thumb instruction, PC reads as the address of the current instruction plus 4.
- Writing an address to PC causes a branch to that address.

In Thumb code, most instructions cannot access PC.

ARMv7 Reference Manual at A2-11 (43) [TPLBN049331]; *see also* B1-9 (1159) [TPLBN050447], ARMv6 Reference Manual at A2-36 (36) [TPLBN035882], B1-211 (211) [TPLBN036057], B1-216 – B1-217 (216-217) [TPLBN036062-63].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

	PC Program Counter, see *Use of 0b1111 as a register specifier* on page A5-82 for more information. The PC is loaded with the Reset handler start address on reset. PC is sometimes referred to as R15. ARMv6 Reference Manual at A2-36 (36) [TPLBN035882]; *see also* ARMv7 Reference Manual at A2-11 (43) [TPLBN049331].
[43b] said fetching means configured to locate the multiple sequential instructions using the address bits from the program counter.	On information and belief, each Accused Product includes a fetching means configured to locate the multiple sequential instructions using the address bits from the program counter. As discussed above in connection with claim 1, on information and belief, each Accused Product includes a means connected to the bus for fetching instructions for the central processing unit integrated circuit on the bus from the memory. *See* element 1d, above. As further discussed above, the means for fetching instructions is configured and connected to fetch multiple sequential instructions from the memory in parallel and supply the multiple sequential instructions to the central processing unit integrated circuit during a single memory cycle. *See* element 1e, above. As further discussed above, the means for fetching instructions is configured and connected to fetch multiple sequential instructions from said memory in parallel and supply the multiple sequential instructions to the central processing unit integrated circuit during a single memory cycle comprises supplying the multiple sequential instructions in parallel to the instruction register during the same memory cycle in which the multiple sequential instructions are fetched. *See* element 1L, above. As discussed above in connection with element 43a, the Accused Products all include a program counter that is stores address bits during operation. *See* element 43a, above. In addition, the documents cited above show that the addresses stored by the program counter are addresses of instructions. The program counters in the Accused Products are used to store the address of the current instruction and used when executing a program/subroutine.

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

LR, the Link Register

Register R14 is used to store the return address from a subroutine. At other times, LR can be used for other purposes.

When a BL or BLX instruction performs a subroutine call, LR is set to the subroutine return address. To perform a subroutine return, copy LR back to the program counter. This is typically done in one of two ways, after entering the subroutine with a BL or BLX instruction:

- Return with a `BX LR` instruction.
- On subroutine entry, store LR to the stack with an instruction of the form:
  `PUSH {<registers>,LR}`
  and use a matching instruction to return:
  `POP {<registers>,PC}`

ThumbEE checks and handler calls use LR in a similar way. For details see Chapter A9 *ThumbEE*.

PC, the Program Counter

Register R15 is the program counter:

- When executing an ARM instruction, PC reads as the address of the current instruction plus 8.
- When executing a Thumb instruction, PC reads as the address of the current instruction plus 4.
- Writing an address to PC causes a branch to that address.

In Thumb code, most instructions cannot access PC.

ARMv7 Reference Manual at A2-11 (43) [TPLBN049331]; *see also* B1-9 (1159) [TPLBN050447], ARMv6 Reference Manual at A2-36 (36) [TPLBN035882], B1-211 (211) [TPLBN036057], B1-216 – B1-217 (216-217) [TPLBN036062-63].

EXHIBIT E-3 – CLAIM CHART FOR INFRINGEMENT OF U.S. PATENT NO. 5,440,749 By Samsung

Claim 59	
[59 preamble] The microprocessor system of claim 9	As shown in connection with claim 9, on information and belief, each Accused Product contains one of the Accused Microprocessors and includes all elements of claim 9. *See* claim chart for claim 9, above.; *see also* claim chart for claim 1, above.
[59a] wherein the microprocessor system comprises an instruction register configured to store the multiple sequential instructions and from which instructions are accessed and decoded; and	On information and belief, each Accused Product includes an instruction register configured to store the multiple sequential instructions and from which instructions are accessed and decoded. As discussed above in connection with claim 1, on information and belief, each Accused Product includes an instruction register configured to store the multiple sequential instructions from which instructions are accessed and decoded. *See* element 1k, above.
[59b] wherein the means for fetching instructions being configured and connected to fetch multiple	On information and belief, each Accused Product includes a means for fetching instructions that is configured and connected to fetch multiple sequential instructions from memory in parallel and supply the multiple sequential instructions to the central processing unit during a single memory cycle comprises supplying the multiple sequential instructions in parallel to the instruction register during the same memory cycle in which the multiple sequential instructions are fetched As discussed above in connection with claim 1, on information and belief, each Accused Product includes the claimed means for fetching. *See* element 1L, above.

sequential instructions from said memory in parallel and supply the multiple sequential instructions to the central processing unit during a single memory cycle comprises supplying the multiple sequential instructions in parallel to said instruction register during the same memory cycle in which the multiple sequential instructions are fetched.	