FTF FREESCALE TECHNOLOGY FORUM 2015

i.MX 6SoloX Heterogeneous Multiprocessing with ARM® Cortex®-A9 + Cortex-M4 Architecture

FTF-DES-F1132

Glen Wienecke | i.MX Systems Architect

J U N E . 2 0 1 5

freescale™

External Use

Agenda

- Key Features of the i.MX 6SoloX Heterogeneous MultiProcessing (HMP) Architecture
- Concepts of Resource Domains for System Partitioning
- Introduction of Resource Domain Controller (RDC)
- HMP InterProcessor Communication (IPC)
- HMP Power Domain Partitioning
- HMP Enablement in Linux/MQX BSP

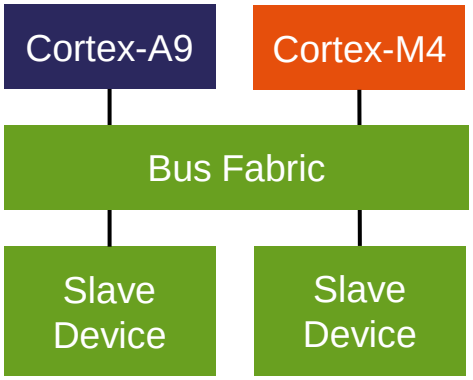
- Key Features of the i.MX 6SoloX Heterogeneous MultiProcessing (HMP) Architecture
- Concepts of Resource Domains for System Partitioning
- Introduction of Resource Domain Controller (RDC)
- HMP InterProcessor Communication (IPC)
- HMP Power Domain Partitioning
- HMP Enablement in Linux/MQX BSP

i.MX 6SoloX HMP Key Features

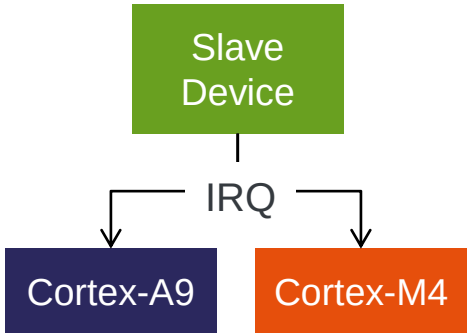
Feature	HMP Benefits
Integration of Cortex-A9 and Cortex-M4 processors	• Execute rich OS on Cortex-A9 and real-time software on Cortex-M4 • Cortex-M4 enhances low-power capability • Use Cortex-M4 to increase system integrity and security • Leverage proven Cortex-M4 software solutions
Shared Resource Architecture	• Efficient use of system resources • Flexibility to adapt to new use cases • High-performance HMP communication via shared memory • Memory map leverages Cortex-M4 architectural features
Resource Domain Controller	• Allows software to partition system resources into domains with assignable access permissions • Integrated hardware semaphore facilitates safe sharing of peripherals
Messaging Unit (MU)	Flexible interprocessor communication
Hardware Semaphore (SEMA4)	HMP synchronization to shared resources
Power Domain Partitioning	Flexible power domain partitioning to enable low-power processing

HMP Shared Resource Architecture

Shared Bus Topology



Shared Peripheral Interrupts



- Shared Bus Topology
  - Allows repartitioning of resources for new use cases
  - Memory devices can be shared between cores to reduce BOM
  - Shared memory allows high-performance HMP communication
- Shared Peripheral Interrupts
  - Interrupt requests routed to both cores
  - Masking is implemented using interrupt controllers inside cores

HMP Memory Map — Internal Memory

- Architectural differences between ARMv7-A and ARMv7-M prevent memory maps of Cortex-A9 and Cortex-M4 from being identical
- Internal On-Chip RAMs (OCRAM, OCRAM_S, OCRAM_L2) are accessible in multiple views of the Cortex-M4
  - Cortex-M4 code bus view (0x008F_8000 — 0x009F_FFFF)
    - Leverages Cortex-M4 code bus and associated code bus cache
    - ARMv7-M default cache policy for this region is write-through
    - Avoids address translations (same addressing as Cortex-A9)
    - Avoids extra cycle of instruction/vector fetches from Cortex-M4 system bus
  - Cortex-M4 system bus view (0x208F_8000 — 0x209F_FFFF)
    - Leverages Cortex-M4 system bus and associated system bus cache
    - ARMv7-M default cache policy for this region is write-back, write-allocate
    - Data fetches from Cortex-M4 system bus will not incur extra cycle

HMP Memory Map — External Memory

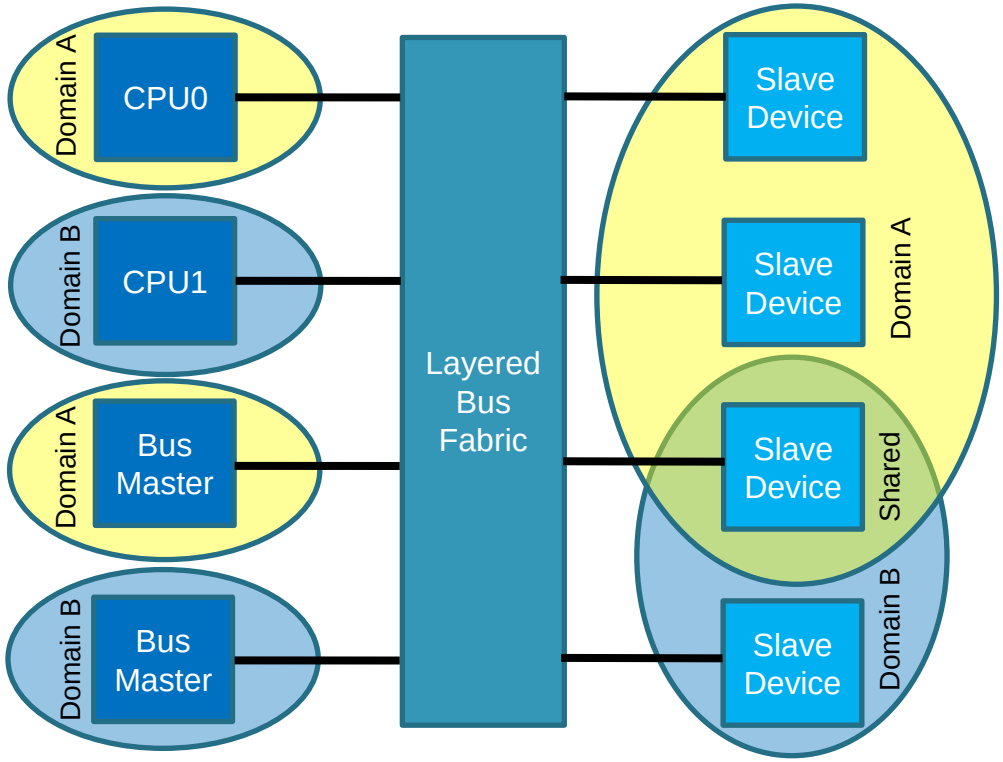
- External Memories (EIM, QSPI, DRAM) are accessible in multiple views of the Cortex-M4
  - Cortex-M4 code bus view (0x0400_0000 — 0x1FF7_FFF)
    - Leverages Cortex-M4 code bus and associated code bus cache
    - ARMv7-M default cache policy for this region is write-through
    - Avoids extra cycle of instruction/vector fetches from Cortex-M4 system bus
    - Partial mapping of external memory (Example: 64 MB of 256 MB for QSPI1)
  - Cortex-M4 system bus view (0x5000_0000 — 0xDFFF_FFFF)
    - Leverages Cortex-M4 system bus and associated system bus cache
    - Avoids address translations (same addressing as Cortex-A9)
    - Data fetches from Cortex-M4 system bus will not incur extra cycle
    - Full mapping of EIM and QSPI memories
    - Partial mapping of DRAM (1.5 GB) due to Cortex-M4 system address space

Agenda

- Key Features of the i.MX 6SoloX Heterogeneous MultiProcessing (HMP) Architecture
- Concepts of Resource Domains for System Partitioning
- Introduction of Resource Domain Controller (RDC)
- HMP InterProcessor Communication (IPC)
- HMP Power Domain Partitioning
- HMP Enablement in Linux/MQX BSP

Resource Domains

- Use ***resource domains*** to partition the system
  - Masters are assigned to a resource domain
  - Slave access permissions are defined per resource domain
  - Memory region access permissions are defined per resource domain
- Sideband signals of bus fabrics carry resource domain ID

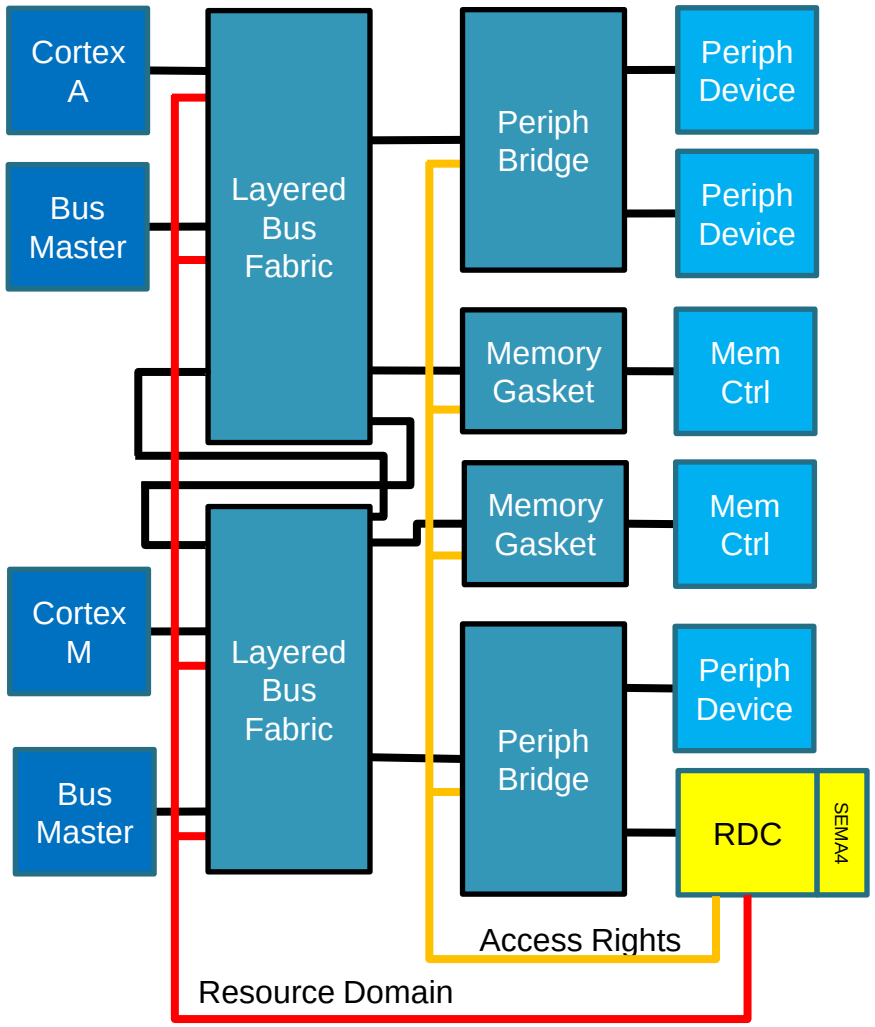


Agenda

- Key Features of the i.MX 6SoloX Heterogeneous MultiProcessing (HMP) Architecture
- Concepts of Resource Domains for System Partitioning
- Introduction of Resource Domain Controller (RDC)
- HMP InterProcessor Communication (IPC)
- HMP Power Domain Partitioning
- HMP Enablement in Linux/MQX BSP

Resource Domain Controller (RDC)

- Resource Domain Controller (RDC) is a new module integrated into next-gen i.MX devices
- RDC provides a centralized programming model to configure isolation and sharing of system resources
- **Key RDC features:**
  - Assignment of master resources (CPUs and bus mastering peripherals) to a ***resource domain***
  - Configuration of read/write access for slave peripherals based on ***resource domain***
  - Partitioning of memory into regions that can have separate domain access controls
  - Configuration of read/write access for memory regions based on ***resource domain***
  - Integral semaphore hardware enables cooperative software to safely access peripherals with access by multiple domains
  - Optional enforcement of semaphore usage to reject accesses by master resources that have not obtained the semaphore lock



i.MX 6SoloX RDC Configuration

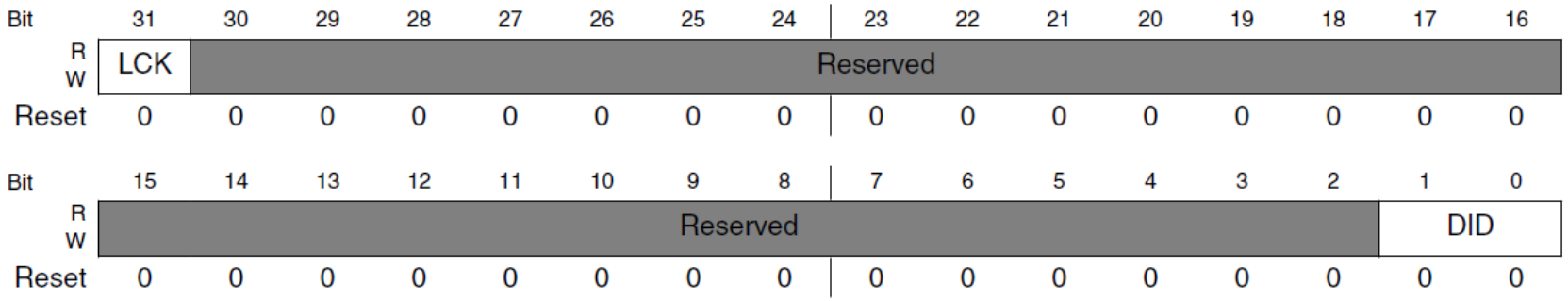
- 4 resource domains (RDC IP can support up to 16)
- 32 masters with assignable domains (27 defined for i.MX 6SoloX)
- 110 peripherals (each AIPS peripheral has domain access control register and hardware semaphore)
- Total of 55 memory regions with the following configuration

Memory/Port	Number of Regions	Region Resolution
MMCD	8	4 KB
QSPI1 (read path)	8	4 KB
QSPI2 (read path)	8	4 KB
WEIM	8	4 KB
PCIe	8	4 KB
OCRAM	5	128 B
OCRAM_S	5	128 B
OCRAM_L2	5	128 B

RDC Programming Model

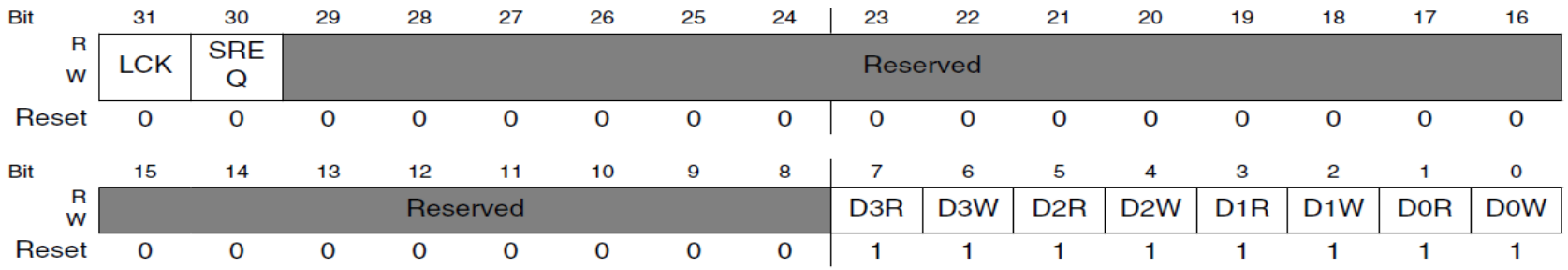
- MDA (Master Domain Assignment) registers allow software to assign a resource domain to each master in the system
- PDAP (Peripheral Domain Access Permissions) registers allow software to specify which resource domains are allowed access the peripheral
- MRSA (Memory Region Start Address), MREA (Memory Region End Address), and MRC (Memory Region Control) registers allow software to partition memory interfaces into regions and specify which resource domains are allowed to access each region
- SEMAPHORE registers work in concert with the PDAP to optionally force software to acquire a hardware semaphore before accessing a shared peripheral

Master Domain Assignment (MDA)



- LCK — Used to lock configuration and prevent further modification until SoC is reset
- DID — Resource domain (4 domains supported) to which a master is assigned

Peripheral Domain Access Permissions (PDAP)

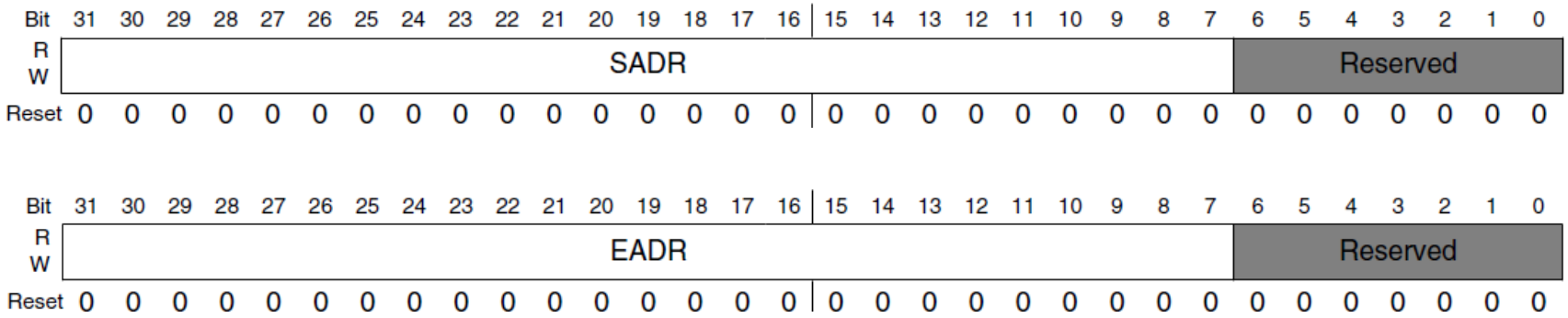


Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LCK	SREQ	Reserved													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								D3R	D3W	D2R	D2W	D1R	D1W	D0R	D0W
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

- LCK — Used to lock configuration and prevent further modification until SoC is reset
- SREQ — Used to specify that a semaphore will be required when accessing a shared peripheral
- DxR — Read access permission for each domain
- DxW — Write access permission for each domain

Memory Region Start/End Address (MRSA/MREA)

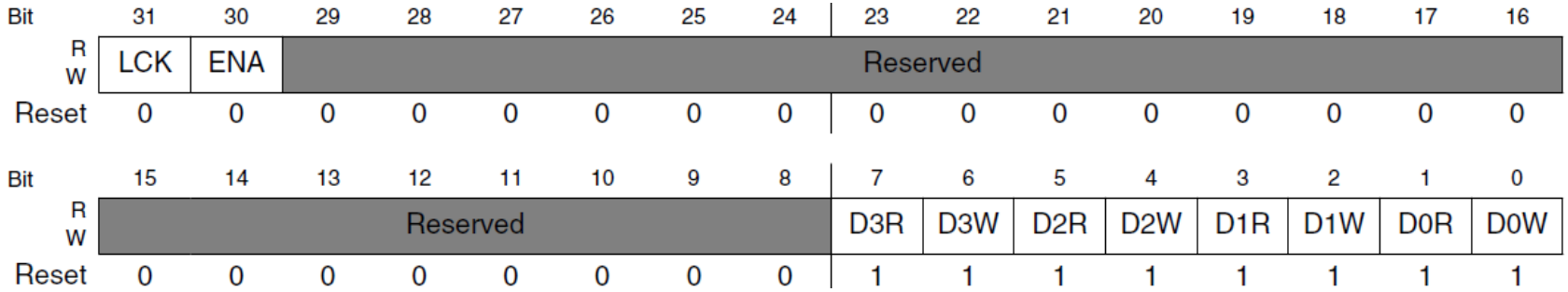


- SADR — Start address for memory region
- EADR — End address (exclusive) for memory region

Note: Reserved bits will vary depending on granularity of respective region. Register description shown is for 128-byte granularity.

Note: Access permissions are prioritized for overlapping regions. Lower order entries in the memory region table have precedence.

Memory Region Control (MRC)

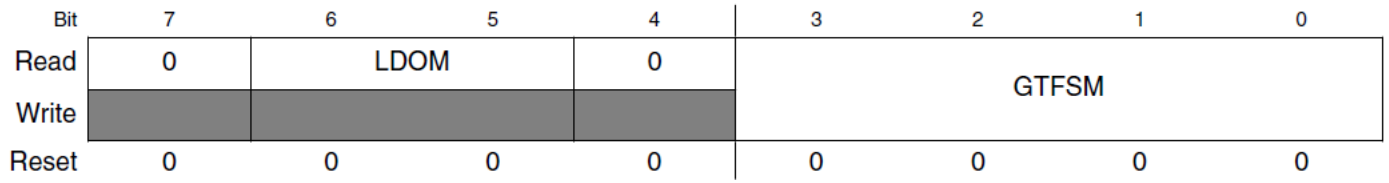


Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LCK	ENA	Reserved													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								D3R	D3W	D2R	D2W	D1R	D1W	D0R	D0W
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

- LCK — Used to lock configuration and prevent further modification until SoC is reset (exception is ENA bit which can be set but not cleared after LCK is set)
- ENA — Enables the memory region access controls (useful to defer enabling regions until all have been configured)
- DxR — Read access permission for each domain
- DxW — Write access permission for each domain

SEMAPHORE_GATE



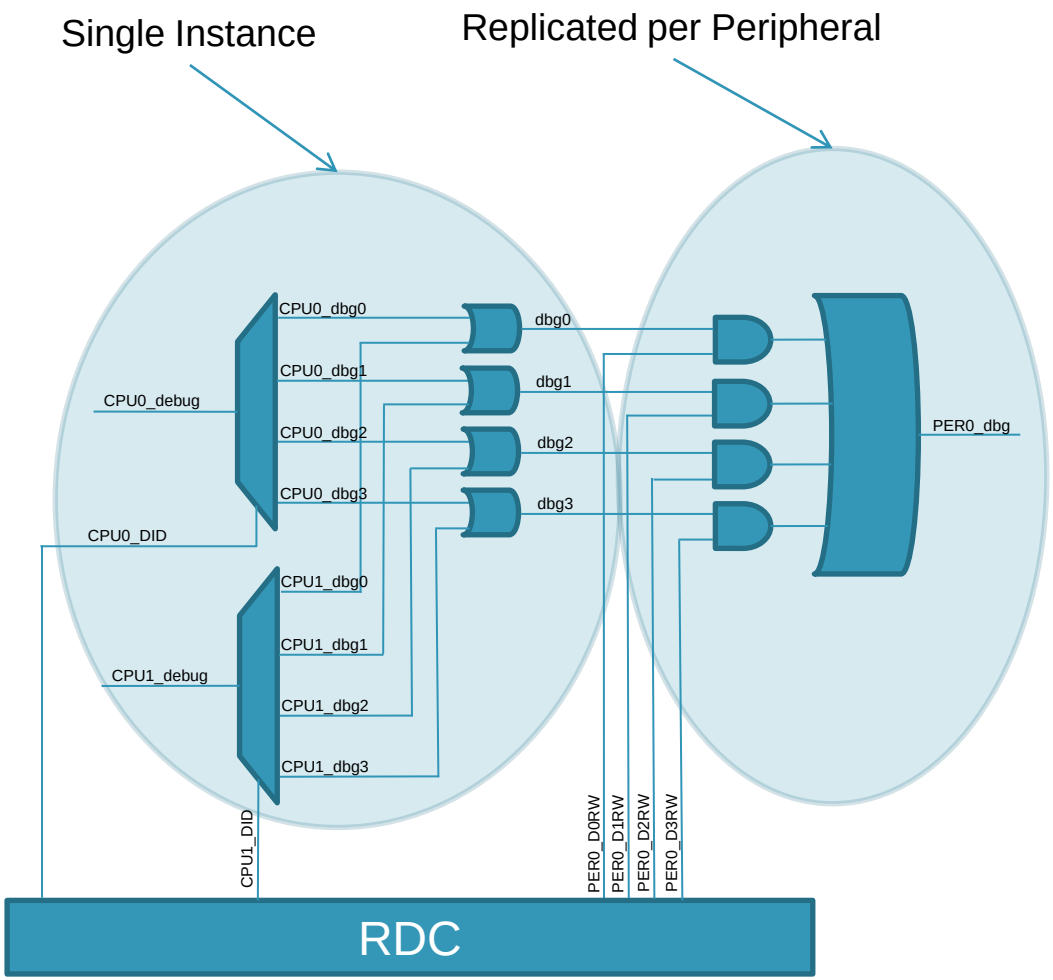
- LDOM — Indicates which domain currently has locked the gate
- GTFSM — State of hardware gate

Note: The GTFSM must be written with the master index (masterID) + 1 as documented in the RM. Attempts to lock the gate by a master that does not specify its own masterID will be ignored by the state machine. Attempts to unlock the gate by a master that does not currently own the lock will be ignored. The SREQ (require semaphore logic) can be used to enforce masters to properly acquire the semaphore for shared peripherals.

RDC Support for Low-Power Modes

- Power gating of domains that contain memory gaskets causes region-based protection information to be lost
- Bus masters in domains that remain powered have the ability to generate accesses immediately after power is restored to domains that contain memory gaskets
- Security may be compromised after domain power is restored until the region-based protection is reloaded into the gaskets
- RDC provides logic to save the region-based configuration and restore the settings after power to the impacted domain is restored
- RDC prohibits access to the memories during low-power modes under the following conditions:
  - Gasket is powered off due to domain power gating
  - Gasket configuration is being restored after domain power gating
- Software can poll or receive an interrupt to be notified when the gasket configuration restoration is complete

RDC Control of Debug Signaling



- A simple logical OR of CPU debug signals can cause problems during system debug
- RDC access permissions are used to control peripheral debug behavior
- CPU debug signals generate domain-specific debug signals based on domain ID set in RDC
- Domain-specific debug signals from CPUs are logically ORed together to create system-wide, domain-specific debug signals
- For each peripheral, the system-wide, domain-specific debug signals will be “qualified” with the corresponding domain read/write permissions set in the RDC
- The qualified domain-specific debug signals are then logically ORed to create a peripheral-specific debug signal

RDC Initialization

- RDC should be isolated to ensure that only a trustworthy master resource can configure the registers
- Recommended options for initialization of RDC:
  - Configure RDC during secure boot and lock configuration registers from further modification
  - Configure the RDC to be accessible only from CA9 and use CSU to further restrict access to secure supervisor software (TrustZone)
  - Configure the RDC to be accessible only from a trustworthy domain and use CSU to further restrict access based on security level

Agenda

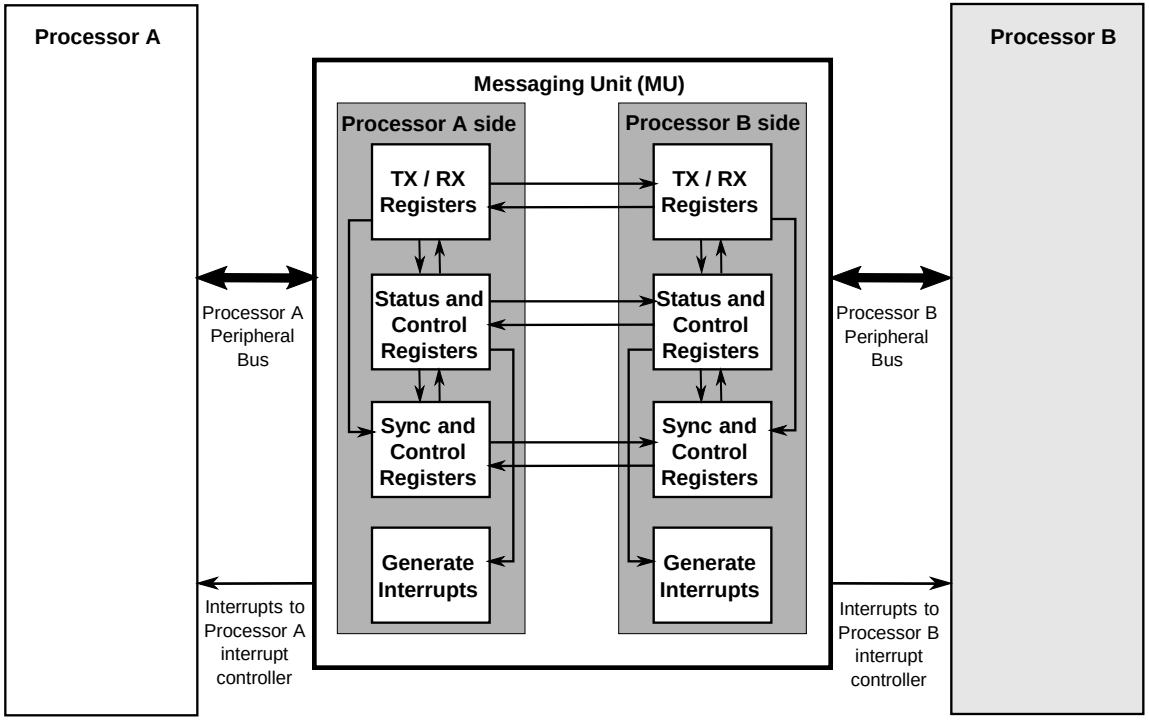
- Key Features of the i.MX 6SoloX Heterogeneous MultiProcessing (HMP) Architecture
- Concepts of Resource Domains for System Partitioning
- Introduction of Resource Domain Controller (RDC)
- HMP InterProcessor Communication (IPC)
- HMP Power Domain Partitioning
- HMP Enablement in Linux/MQX BSP

IPC Hardware Summary

Hardware	Features
Messaging Unit (MU)	Mailbox registers to send/receive messages Provided interprocessor interrupts
SEMA4	Hardware-based general-purpose semaphore module
Shared Memory	Bus topology allows shared memory RDC and CSU can provide memory protection/isolation
Exclusive Access	ARMv7-A and ARMv7-M defines exclusive access instructions (LDREX/STREX)

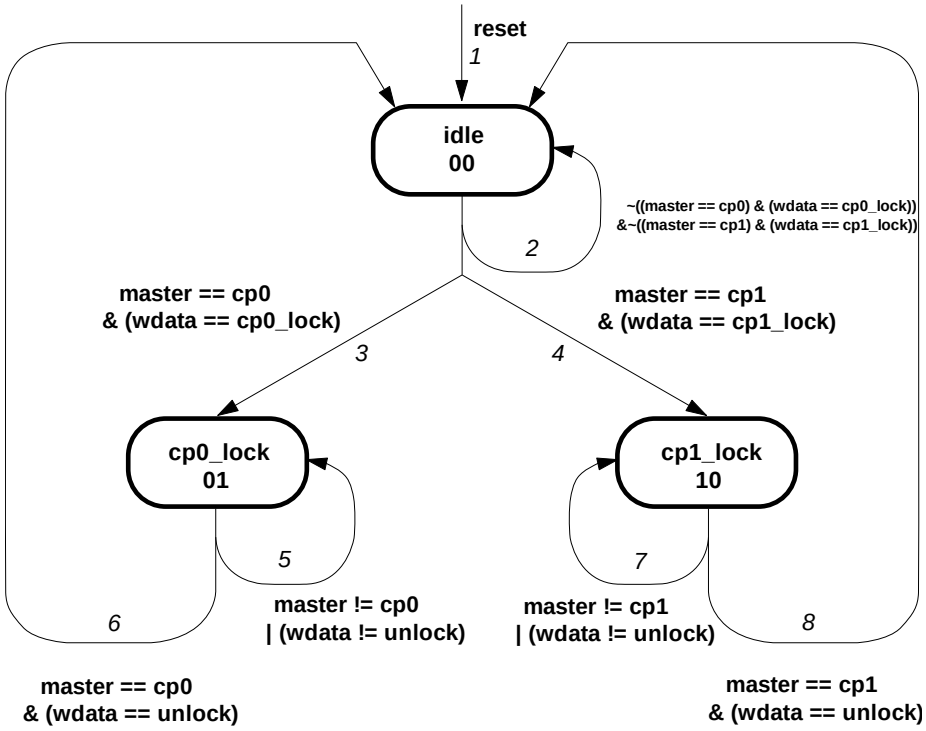
Messaging Unit (MU)

- Proven IP from cellular baseband SoCs
- Messaging control by interrupts or polling
- 4 RX/TX registers on each side
- 12 interrupt requests (IRQs) per side
  - 4 RX register full IRQs
  - 4 TX register empty IRQs
  - 4 general-purpose IRQs
- 3 general-purpose flags per side



Semaphore (SEMA4)

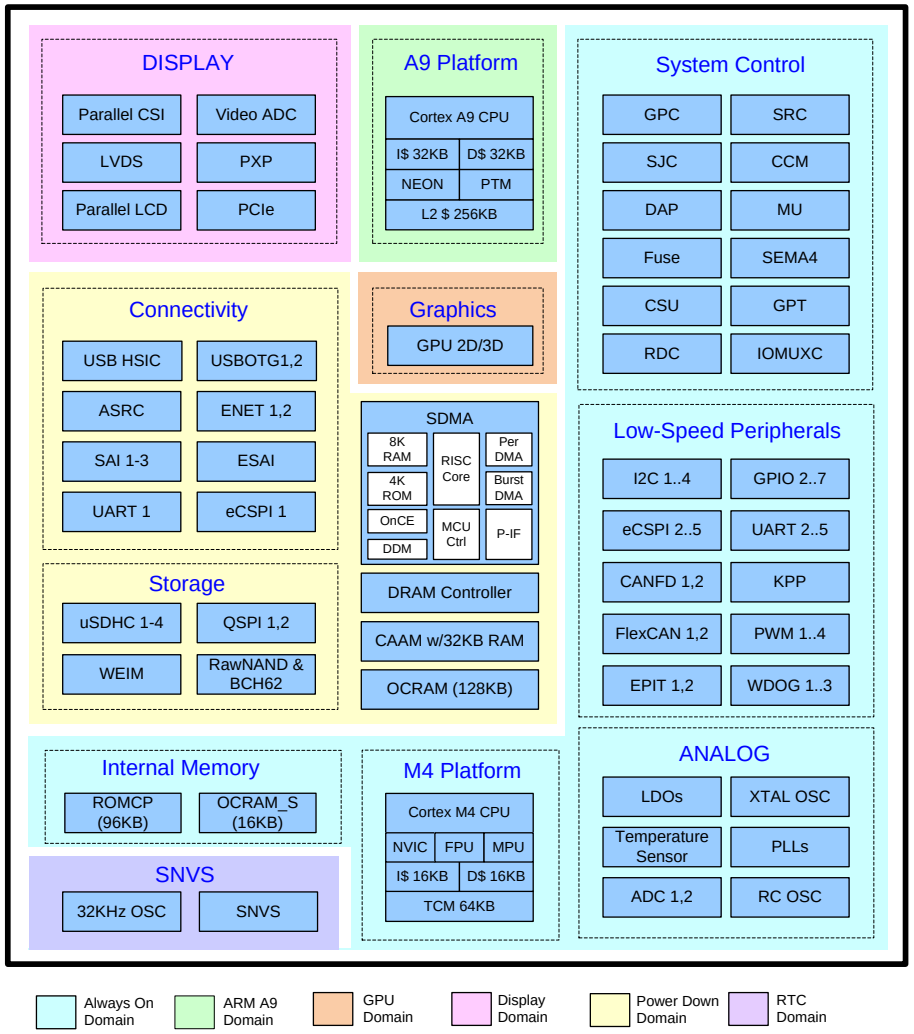
- Provides a hardware mechanism for cooperative software to safely share resources in HMP systems
- Module reused from Vybrid and Freescale automotive MCUs
- Separate module from RDC semaphore
- Supports 16 general-purpose hardware semaphores
- Semaphore can only be unlocked by locking processor
- Optional interrupt notification after failed lock attempt to indicate when semaphore is unlocked
- Software conventions still required to ensure only processor with semaphore lock can access shared resources



Agenda

- Key Features of the i.MX 6SoloX Heterogeneous MultiProcessing (HMP) Architecture
- Concepts of Resource Domains for System Partitioning
- Introduction of Resource Domain Controller (RDC)
- HMP InterProcessor Communication (IPC)
- HMP Power Domain Partitioning
- HMP Enablement in Linux/MQX BSP

Power Domain Partitioning



- SoC is divided into 6 power domains for flexible power gating (not including analog domains)
- Cortex-M4 and low-power peripherals are located in a separate low-leakage domain (Always On Domain) to enable low-power processing

Agenda

- Key Features of the i.MX 6SoloX Heterogeneous MultiProcessing (HMP) Architecture
- Concepts of Resource Domains for System Partitioning
- Introduction of Resource Domain Controller (RDC)
- HMP InterProcessor Communication (IPC)
- HMP Power Domain Partitioning
- HMP Enablement in Linux/MQX BSP

RDC Support in Linux

- U-Boot RDC driver supports system partitioning of masters and peripherals into resource domains
  - `arch/arm/imx-common/rdc-sema.c`.
- Peripherals configured as shared will also be configured with SREQ (requires semaphore) bit set
- Customers must call lock/unlock APIs
  - `arch/arm/include/asm/imx-common/rdc-sema.h`
- GPIO1 is set in U-Boot to be shared (refer to shared resources array)
  - `/freescale/mx6sxsabresd/mx6sxsabresd.c`
- Example of a driver involved with RDC usage is `mxc_gpio.c`
- Memory region partitioning is not supported

RDC Support in MQX

- MQX configures the RDC for M4 resources
  - `mqx/source/bsp/imx6sx_sdb_m4/init_hw.c`
- RDC PDAP registers are configured for M4-only access to the following peripherals: UART2, I2C3, ECSPI4, ECSPI5, ADC1, ADC2, CAN1, CAN2, CANFD_CAN1, CANFD_CAN2, EPIT1, EPIT2, WDOG3
- RDC memory region registers are configured for M4-only access to 256KB of QSPI XIP space starting at 0x78000000

IPC Feature Support in Linux/MQX

- No standalone MU driver
- MU exposed in MCC (MultiCore Communication)
  - `/arch/arm/mach-imx/mcc_linux.c`
  - `/arch/arm/mach-imx/mcc_imx6sx.c`
- SEMA4 driver supports lock/unlock APIs
  - `/drivers/char/imx_mcc/imx_sema4.c`

- HMP power management leverages IPC hardware
  - SEMA4 used to synchronize accesses to shared PM data structures
  - MU used to communicate PM requests/acknowledgements between the execution environments

www.Freescale.com