ARM Advanced Interrupt Controller

Considering AT91SAM7X256 Device

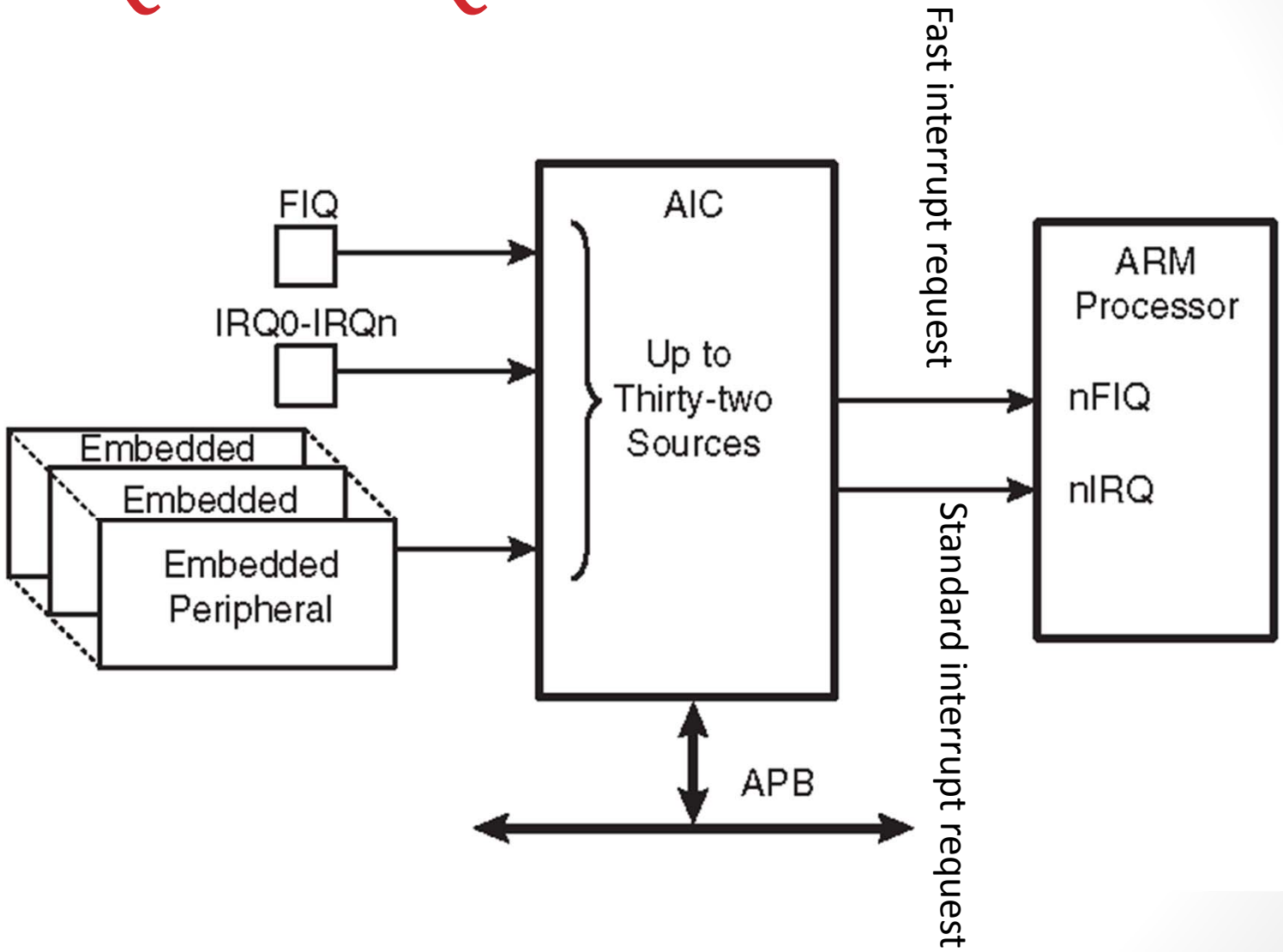
Microcontrollers and Microprocessors Course

Isfahan University of Technology – Dec 2010

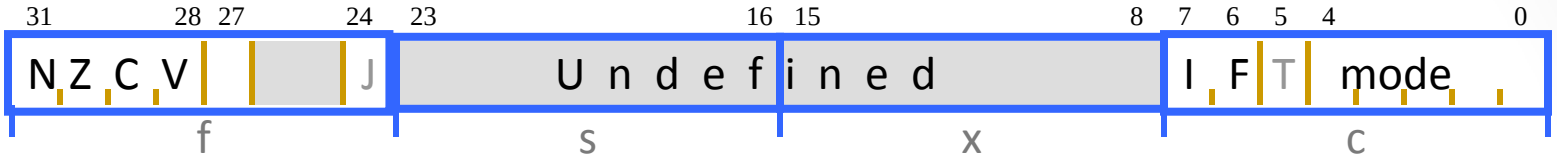
Advanced Interrupt Controller (AIC)

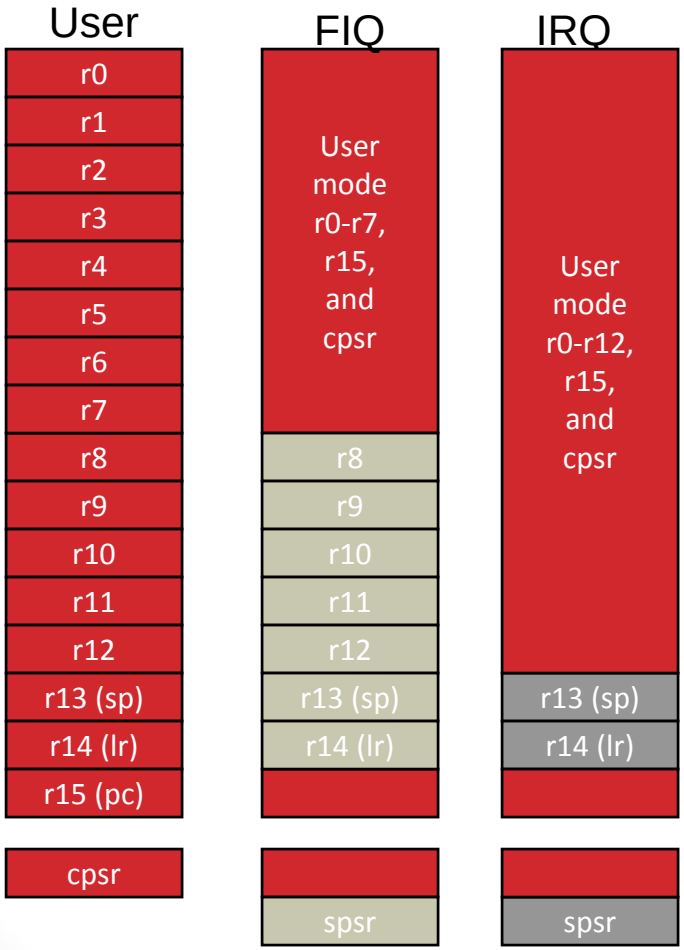
- AIC used in ATMEL device
  - Not necessarily the same between all manufacturers
  - Concepts are the same
- 8 Levels of priorities
  - 7 highest priority
  - 0 lowest
- Maskable
  - You can enable/disable each interrupt individually
- Interrupt Vector Table
  - Interrupt service routine can be any where!
  - You store its address in Interrupt vector table

nFIQ & nIRQ


FIQ
IRQ0-IRQn
Embedded
Embedded
Embedded
Peripheral
AIC
Up to
Thirty-two
Sources
APB
ARM
Processor
nFIQ
nIRQ
Fast interrupt request
Standard interrupt request

Interrupt Modes





- Interrupt Disable bits.
  - I = 1: Disables the IRQ.
  - F = 1: Disables the FIQ.
- Mode bits
  - Specify the processor mode

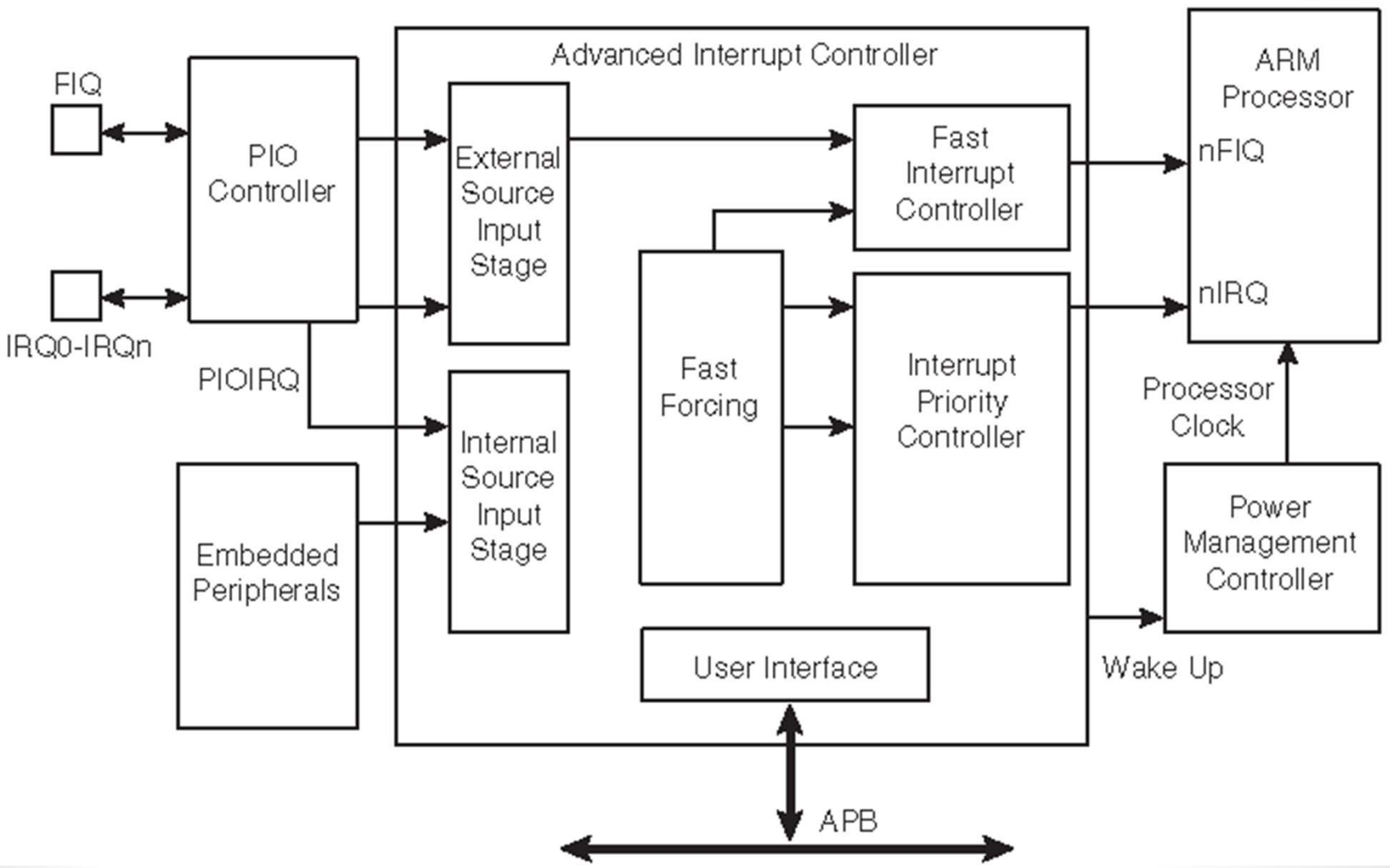
Interrupt Priority

- When two interrupts have happened
  - Pending
  - CPU should serve them
  - CPU serves higher priority interrupt first
- While an interrupt is being served
  - Another interrupt happens
  - If it is lower priority
    - It should wait until the ISR finish
  - If higher priority:
    - It will be served in the middle of the previous ISR

Power Management

- AIC is always working
- Power Management Controller has no effect on it!
- ARM core can go to sleep itself
  - AIC can wake up ARM core
    - Producing an interrupt
    - Telling PMC

AIC Block Diagram



Interrupt Sources

- 0 : FIQ
  - If microcontroller has no FIQ pin it is left unused
- 1 (SYS): System interrupt
  - CAN BE the result of all system peripherals
    - Timer
    - Real Time Clock
    - Power Management Controller
    - Memory Controller
    - ….
  - Or it can be dedicated to one peripheral ! (for example timer)
- 2-31 (PID2-PID31): can be used
  - by peripherals or
  - as external interrupt

AIC Source Mode Register

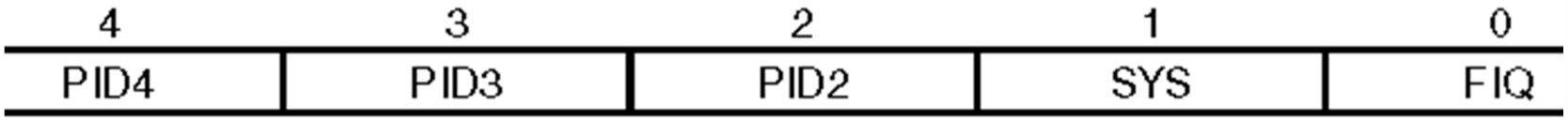
- AIC_SMR
  - 32 Registers for 32 interrupts
- Modes
  - Triggered : cleared automatically
  - Level : should be clear by ISR
  - High/Low : for internal interrupt just high

5	4	3	2 1
SRCTYPE	–	–	PRIOR

SRCTYPE		Internal Interrupt Sources	External Interrupt Sources
0	0	High level Sensitive	Low level Sensitive
0	1	Positive edge triggered	Negative edge triggered
1	0	High level Sensitive	High level Sensitive
1	1	Positive edge triggered	Positive edge triggered

Interrupt Enable/Disable

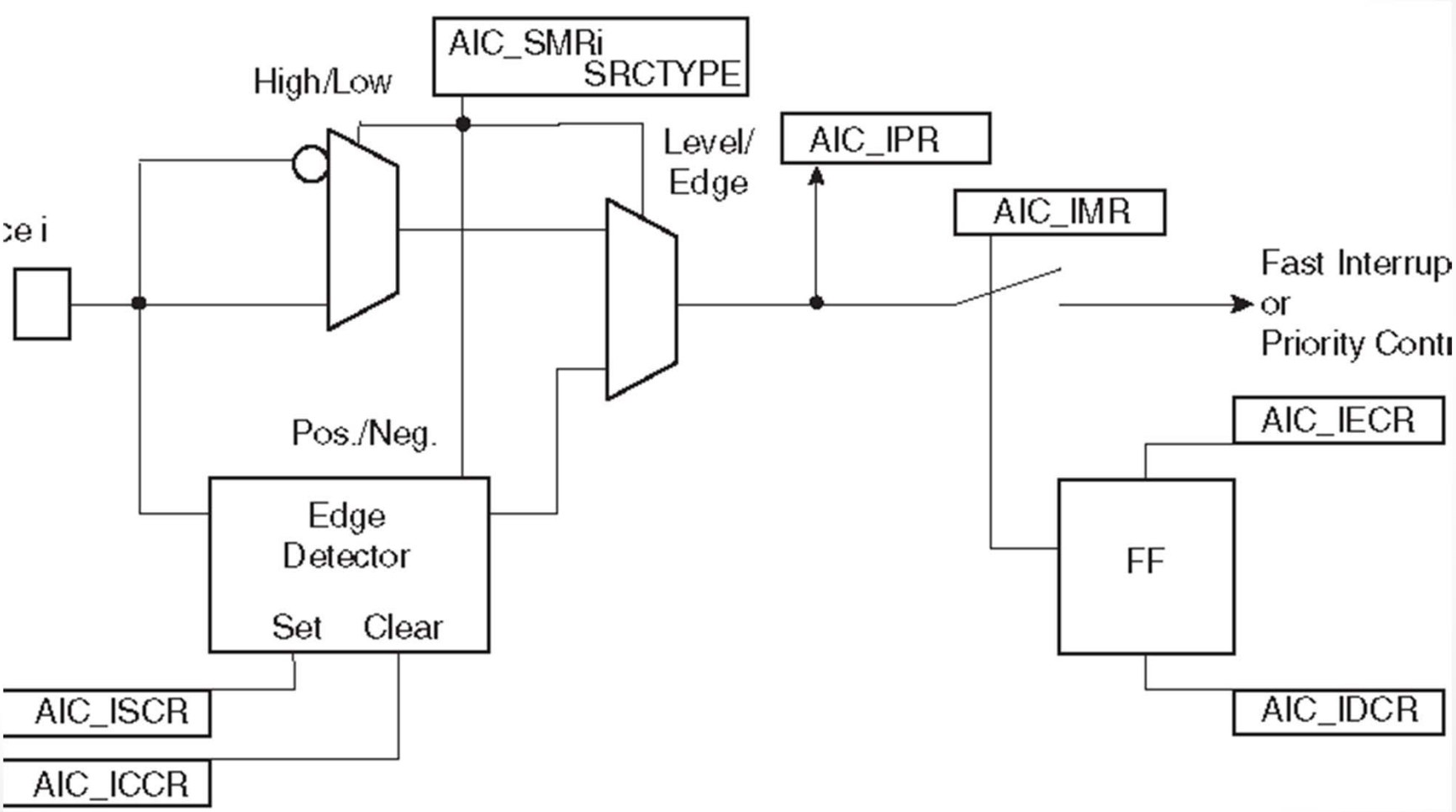
- Interrupt Enable Command Register : AIC_IECR
- Interrupt Disable Command Register : AIC_IDCR
- Interrupt Mask can be read in : AIC_IMR



Interrupt pending & status

- Interrupt pending register (AIC_APR)
  - Contains a register
  - Holds all of the interrupts that should occur
- AIC Priority Controller acts on Pending Register
  - Selects interrupt with higher priority
- AIC Interrupt Status Register (AIC_ISR)
  - Contains the number of selected interrupt

Block Diagram



Interrupt Vector Register

- When CPU wants to serve an interrupt
- CPU reads AIC_IVR
- AIC_IVR contains an address
- Address indicates the location of ISR
- This address is mainly held by: AIC_SVR (Source Vector Register)
  - AIC_SVR is an array: 32 elements
  - AIC_IVR = AIC_SVR[Current Interrupt Number]
- CPU jumps to ISR
- nIRQ will no more get active
  - Until ISR write to AIC_EOICR
  - Exception : Interrupts with higher priority

Interrupt Nesting

- A dedicated 8 level stack is available
- For serving interrupts with higher priorities
  - Inside another ISR

Intro to one of AT91SAM7X256 peripherals

PERIODIC INTERVAL TIMER

Periodic Interval Timer

- Generates periodic interrupts
  - Mainly used in OS Scheduler
- Two counters:
  - 20 Bit CPIV
  - 12 Bit PICNT
  - Clock : Master Clock/16
- Operation:
  - CPIV counts upward
  - From 0 to PIV
  - An interrupt occurs (PITIEN in PIT_MR should be active)
    - PICNT increments by 1
    - PITS in Status Register (PIT_SR) gets 1
- PIV is 20 Bits value
  - In PIT Mode Register (PIT_MR)

Periodic Interval Value Register

- PIT_PIVR contains
  - CPIV
  - PICNT
- When you read PIVR
  - CPIV does not change : it continues counting
  - PICNT sets to zero
  - PITS sets to zero
- Periodic Interval Image Register
  - PIT_PIIR
  - Nothing changes

Block Diagram