SNS COLLEGE OF TECHNOLOGY

(An Autonomous Institution)

COIMBATORE-35

Accredited by NBA-AICTE and Accredited by NAAC – UGC with A++ Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

COURSE NAME: 19EEB303 / Microcontroller and its Applications

III YEAR / VI SEMESTER

Unit III – **IOT - ARCHITECTURE REFERENCE MODEL**

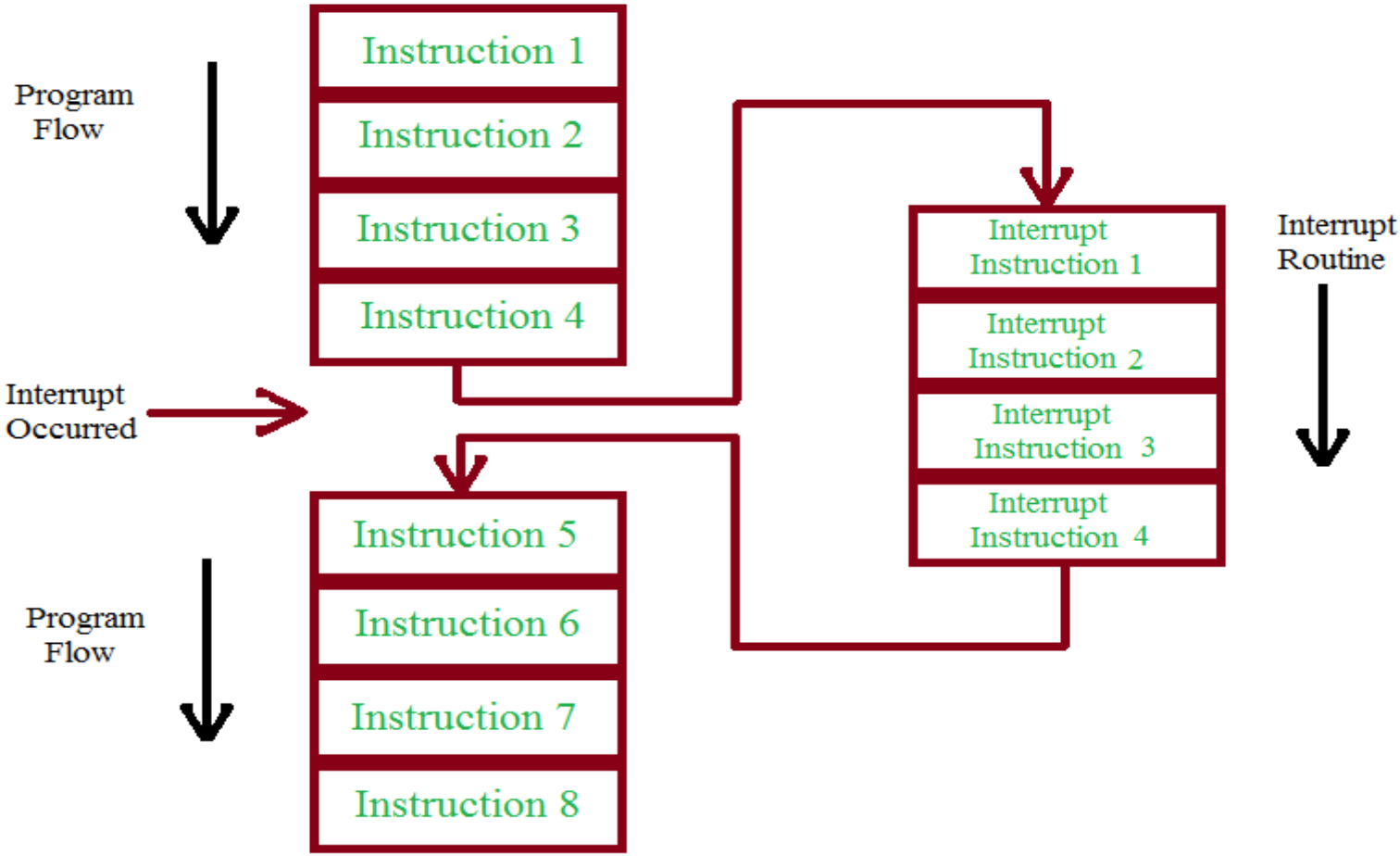
Topic:Interrupts

Interrupts

The main purpose of any microcontroller is to accept input from input devices and accordingly drive the output. Hence, there will be several devices connected to a microcontroller at a time. Also, there are many internal components in a microcontroller like timers, counters etc. that require attention of the processor.

Since all the devices can't obtain the attention of the processor at all times, the concept of "Interrupts" comes in to picture. An Interrupt, as the name suggests, interrupts the microcontroller from whatever it is doing and draws its attention to perform a special task. The following image depicts the procedure involved in Interrupts.

Interrupts


Program Flow
Instruction 1
Instruction 2
Instruction 3
Instruction 4
Interrupt Occurred
Interrupt Instruction 1
Interrupt Instruction 2
Interrupt Instruction 3
Interrupt Instruction 4
Interrupt Routine
Instruction 5
Instruction 6
Instruction 7
Instruction 8
Program Flow

Interrupts

In the event of an interrupt, the source of the interrupt (like a Timer, Counter etc.) sends a special request to the processor called Interrupt Request (IRQ) in order to run a special piece of code. The special code or function is called as Interrupt Service Routine (ISR).

From the above figure, the CPU executes its normal set of codes until an IRQ occurs. When the IRQ signal is received, the CPU stops executing the regular code and starts executing the ISR. Once the execution of the ISR is completed by the CPU, it returns back to execution of the normal code.

Vectored Interrupt Controller (VIC) handles the interrupts in LPC214x series of MCUs. It can take up to 32 Interrupt Requests. The interrupts in LPC2148 microcontroller are categorized as Fast Interrupt Request (FIQ), Vectored Interrupt Request (IRQ) and Non – Vectored Interrupt Request. All the interrupts in LPC214x have a programmable settings i.e. the priorities of the Interrupts can be dynamically set.

Interrupts

Of the three categories, the FIQ requests have the highest priority, Vectored IRQ requests have the medium priority and non – vectored IRQ requests have the least priority.

When we are talking about “Vectored” and “Non – Vectored” IRQ requests, we are actually talking about the address of the ISR. In case of Vectored IRQ requests, the CPU has a knowledge of the ISR. A special table called Interrupt Vector Table (IVT) contains all the information about the Vectored IRQ. This information can be about the source of the interrupts, ISR address of the IRQ requests etc.

Interrupts

So, each Vectored IRQ has its own unique ISR address. Out of the possible 32 interrupt requests, 16 interrupt requests can be defined as Vectored IRQ. In this 16 slots, any of the 22 interrupts that are available in LPC2148 can be assigned. In the 16 Vectored IRQ slots, slot 0 has the highest priority while slot 16 has the least priority.

In case of Non – Vectored IRQ, as the name itself indicates, the CPU isn't aware of either the source of the Interrupt or the ISR address of the Interrupts. In this case, the CPU must be provided with a default ISR address. For handling Non – Vectored IRQ requests, a special register called "VICDefVectAddr" is available in LPC2148. The address of the default ISR must be given in this register by the user in order to handle the Non – Vectored IRQ requests.

Interrupts

In general, ARM Interrupts are signals from a computer device or a programme within a controller that prompts the main programme to halt and choose what to do next. When an interrupt occurs in ARM, the ISR (Interrupt Service Routine) is called. When an interrupt is received, a component of a programme takes control and performs the activities necessary to service the interrupt.

Interrupts

•When the microcontroller receives an interrupt signal, it interrupts the main programme flow and saves the location of the next instruction (PC) on the stack pointer (SP).
•It jumps to a defined area in memory known as the interrupt vector table, which contains the ISR's address (Interrupt Service Routine). Each interrupt has its own interrupt service routine (ISR). The interrupt vector table contains the address of the interrupt service routine (ISR), to which the microcontroller jumps.
•It begins executing the Interrupt Service Routine until it reaches the subroutine's last instruction, RETI (Return from Interrupt).

Interrupts

•The microcontroller returns to where it left off or was interrupted before after executing the final instruction in the Interrupt Service Routine. And then it pops the top two bytes of the stack into the PC to acquire the programme counter (PC) address from the stack pointer.

•The programme then begins to run from that address and continues to run the main application.

When an external device asserts the processor's IRQ (Interrupt) pin, an interrupt occurs. This can be utilized by an external device to get the processor's attention.

Interrupts

Types of Interrupts

On an ARM processor, there are two types of interrupts. The interrupt produced by external events from hardware peripherals is the first type, whereas the SWI instruction is the second. Because the ARM core only has one FIQ pin, an external interrupt controller is always used so that the system can have multiple interrupt sources that are prioritised by the interrupt controller. The FIQ interrupt is then raised, and the handler determines which of the external interrupts was raised and handles it.

Interrupts

Interrupt Handling

It is up to the system designer to determine which hardware peripherals are capable of generating specific interrupt requests. We may attach numerous external interrupts to one of the ARM interrupt requests and discriminate between them by utilizing an interrupt controller. System designers have chosen a common design for allocating interrupts:

- Normally, SWIs are used to invoke privileged operating system procedures.
- IRQs are often allocated to interruptions that are used on a regular basis, such as periodic timers.
- FIQ is reserved for a single interrupt source that demands a quick reaction time, such as DMA or any other time-critical activity.

Interrupts

Interrupt exception

The CPU sets the computer to a certain memory location when an exception or interrupt occurs. The address is in the vector table, which is a particular address range. The vector table's entries are instructions that lead to specific routines that deal with a specific exception or interrupt. The vector table, a collection of 32-bit words, is stored at memory map location 0x00000000. On certain CPUs, the vector table can be stored at a higher memory location (starting at the offset 0xffff0000). This functionality may be used by operating systems like Linux and Microsoft's embedded solutions.

Interrupts

The processor suspends regular execution and begins loading instructions from the exception vector table when an exception or interrupt occurs.

The vector table.

Exception/interrupt	Shorthand	Address	High address
Reset	RESET	0x00000000	0xffff0000
Undefined instruction	UNDEF	0x00000004	0xffff0004
Software interrupt	SWI	0x00000008	0xffff0008
Prefetch abort	PABT	0x0000000c	0xffff000c
Data abort	DABT	0x00000010	0xffff0010
Reserved	—	0x00000014	0xffff0014
Interrupt request	IRQ	0x00000018	0xffff0018
Fast interrupt request	FIQ	0x0000001c	0xffff001c

Interrupts

• When the CPU is unable to decode an instruction, an undefined instruction vector is utilized.

•When you perform a SWI instruction, the software interrupt vector is triggered. The SWI instruction is commonly used to call an operating system procedure.

• Prefetch abort vector occurs when the processor attempts to fetch an instruction from an address without the correct access permissions. In the decode step, the real abort happens.

•The data abort vector is identical to the prefetch abort vector, except it is raised when an instruction tries to access data memory without the proper permissions.

•External hardware uses an interrupt request vector to interrupt the processor's usual execution flow. Only if IRQs are not disguised in the cpsr can it be increased.

•Fast interrupt request vectors similar to the interrupt request but are reserved for hardware requiring faster response times. It can only be increased if the FIQs in the cpsr are not disguised.

Interrupts