|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Schedule of Technology Based Entrepreneurship Development Programme on Embedded Systems | | | | |
| Faculty : D.Sivaraj, Assistant Professor, ECE Department, PSGCT | | | | |
| S.  No. | Date | Time | No. of Hours | Topics Discussed |
| 1 | 21/03/19  (Thursday) | 10:00am  to 05:00pm | 6 | INTRODUCTION TO EMBEDDED SYSTEMS AND IOT  Introduction to microprocessors and its mandatory peripherals. Basic building blocks of microprocessor based system / mother board / CPU. Introduction to Microcontroller. Microcontroller as a single chip CPU. Introduction to hardware and software modules of embedded systems. Number representation. Representation of fixed and floating point numbers. Positive and Negative Number representation, etc., Understanding embedded systems as 3D and embedded system can do 3C, IoT as 3A and relationship between embedded systems and IoT through 3B.   * 3A is possible through 3B to do 3C using 3D. Understanding Embedded   Systems and IoT as simple as ABCD..  Note: Discussing the concepts of embedded systems and IoT through TV Remote Control  and Printer. |
| 2 | 22/03/19  (Friday) | 10:00am  to 05:00pm | 6 | DESIGN OF HARDWARE & SOFTWARE MODULES FOR EMBEDDED SYSTEM  Understanding Circuit Schematics, Hardware Interfacing Techniques and Programming Techniques.. Project Development Steps using KEIL IDE. Experimenting how to bring out data from a microcontroller through parallel port. LEDs interfacing techniques through current limiting resistors. *Experimenting tail lamp and brake lamp in an automobile with LEDs*. Understanding the need for delay. *Prototyping Junction Warning System through blinking of an LED with Delay Routine.*  Role of on-chip timers in producing time delay. Experimenting different modes of operation of timers to satisfy different time delay requirements. Experimenting the advantages and limitations of using delay routine and timers for delay generation.   * Prototyping Traffic Light Control (TLC) for a four way junction   Note: Learning how to bring out the data from the microcontroller and prototyping its related applications. *Using timer for producing delay reduces the code size, increases execution speed and save power.* |
| 3 | 25/03/19  (Monday) | 10:00am  to 05:00pm | 6 | DISPLAY INTERFACING AND CODE CONVERSION TECHNIQUES  Switching On / Off individual devices with necessary delay. Setting, configuring and labeling of individual bits in an 8-bit parallel port. Programming the same and experimenting in hardware. Hardware(HW): Circuit Schematic for Interfacing 7-segment display. Software(SW): Code Conversion Techniques: Binary to BCD and BCD to 7-segment code conversion. Displaying the numbers from ‘0’ to ‘9’ using 7-segment display.   * Prototyping down count in TLC, Fuel, Air filling in Petrol Bunk, Digital Clock,   Currency Counting, Electronic Weighing Scale, etc.,  Note: Learning how to bring out the data from the microcontroller, and displaying it in 7-segment display in a user friendly way after multiple code conversion. .. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 4 | 27/03/19  (Wednesda y) | 10:00am  to 05:00pm | 6 | KEY BOARD INTERFACING AND PROGRAMMING TECHNIQUES  Understanding the difference between the electrical switch and electronic switch. Learning how a microcontroller identifies the input. HW: Interfacing a switch with microcontroller. SW: Code for identifying the key press. Key bouncing. De-bouncing techniques using delay routine. Prototyping volume up/down in a remote by pressing the up/down key and displaying the increment/decrement values in 7-segment display. Prototyping Entry/Exit system in a Hall/Mall.  Timer as a counter (T0 (P3.4) / T1 (P3.5)) to increment and decrement the count for volume control, entry/exit system.   * Prototyping Volume Control, Channel Control, On/Off control, entry/exit   sensing using switches, etc.,  Note: Learning how to make a microcontroller to identify the key press. *Using counter for counting the number of key press reduces the code size, increases execution speed and save power.* |
| 5 | 28/03/19  (Thursday) | 10:00am  to 05:00pm | 6 | INTERRUPTS AND INTERRUPT SERVICE ROUTINE  Introduction to Interrupts, need for interrupts, types, Interrupt Service Routine (ISR), Addressing Modes, Program Memory, Vector location, Data Memory, Stack, Stack Pointer, Push/Pop for preserving the content during ISR, Register Banks, Context Switching, Interrupt Priority, Interrupt latency, etc.,  Hardware Interrupts; Edge (Falling and Rising) triggered interrupts and Level (Level-Low and Level-High) triggered interrupts.  *Level triggered interrupts* for applications which demand ‘’as long as”. Eg.(i) As long as the brake pedal is pressed the program execution is in ISR which keeps brake light in ON condition. Eg.(ii) As long as the Reset is active high/low, the controller remains in reset condition.  *Edge triggered interrupts* are suitable for applications which demand “counting”. Eg.(1) For volume up/down in TV remote, Pressing up button, produces a transition from 1 to 0 (falling edge). Identifying each edge makes the program execution to jump to its ISR to increase / decrease the volume.  Eg.(2) In an entry/exit system, for each entry/exit, the sensor produces a pulse (transition from 1 to 0 (falling edge) / transition from 0 to 1 (rising edge)). Identifying each edge makes the program execution to jump to its ISR to increment / decrement the count.  Software Interrupts: Software interrupts are generated by the on-chip peripherals. Timer interrupt is a software interrupt, used to produce a periodic time delay (20ms) for display refreshing. Used in multiplexed 7-segment display for display refreshing. Discussed multiplexed 7-segment display, its Interfacing techniques and its advantages. Programming for timer interrupt and experimenting the same in the target machine by displaying the numbers from 000-255.   * Prototyping Volume Control, Channel up/down, entry/exit sensing using   switches and multiplexed 7-segment display. Software interrupt with timer ISR for periodic display refreshing and Hardware edge triggered ISR for incrementing and decrementing the count value.  Note: Learning how to identify the key press using GPIO, counter, hardware interrupts for different applications. *Using ISR for key, will have highest priority to key press.* |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 6 | 29/03/19  (Thursday) | 10:00am  to 05:00pm | 6 | LINEAR CONTROL AND DIRECTION CONTROL  Light Intensity Control, DC Motor Speed Control, Servo Motor Position Control, Volume Control, etc., are the few examples of linear control. For linear control voltage is the prime requirements. Switching between 5V and 0V are Logical Control or ON OFF control which is square / rectangle wave. Controlling the voltage from 0V to 5V is linear control which is triangular / saw tooth wave. Linear control using microcontroller is possible through Pulse Width Modulation (PWM). By varying the duty cycle (on-time) of the pulse the average voltage in the pwm pin will vary. High frequency pwm waveform give more smoothness. Timer along with Compare register are used for producing PWM in a microcontroller. Experimented the linear control by varying the intensity/brightness of the multiplexed 7-segment display.   * Changing the direction of rotation of a DC motor is done by changing the direction of the magnetic field. * Changing the direction of the magnetic field is done by changing the direction   of the current flow in the coil.   * Changing the direction of the current flow is done by changing the polarity. * Changing the polarity is possible through H-Bridge L293D.   Experimented the speed control of DC motor and direction control of DC motor by interfacing a DC motor with microcontroller through L293D H-bridge driver and writing the suitable PWM code in the microcontroller.  .   * Experimented the speed control and light intensity control using PWM and Direction control using H-Bridge.   Note: Learned both the ON/OFF control and Linear Control techniques. |
| 7 | 29/04/19  (Monday) | 10:00am  to 05:00pm | 6 | DIGITAL SIGNAL PROCESSING  Video Signal Processing, Image Processing, Speech signal processing, Audio signal Processing are the types of digital signal processing. Output follows input. Sampling requirement should be Fs >= 2fm for signal processing. Why Sampling? Analog contains infinite values, digital cannot process with infinite, hence sampling. E.g Microphone -> X(t) -> ADC -> X[n] -> DSP h[n] -> y[n] -> DAC -> y(t) -> Loudspeaker.  Convolution is y[n] = x[n] \* h[n]. Digital signal processing is computation technique and doesn’t require any RC components. Instead of using hardware components such as resistors and capacitors to design a filter, in DSP, just by replacing h(n) value +1 to -1 we can change from low pass to high pass filter. So, its cost effective and reduce the size and weight of the device.  MAC – Multiplier Accumulation Unit. Microcontrollers, output doesn’t follow input so by using MAC unit Digital Signal processing can be performed in real-time.  Convolution Filter: [1,1] is LPF and [1,-1] HPF  Audio signal is one dimensional signal. Image signal is two dimensional and video signal is multi-dimensional signal.   * Using signal generator the analog waveform input is given to DSP kit Vi 6416, and running different filters (low pass filter, high pass filter, bandpass filter) program in the kit. * By changing the signal generator output frequency, we could practically see low pass filter, filtering the higher frequencies and visualize in oscilloscope. * By changing the signal generator output frequency, we could practically see high pass filter, filtering the lower frequencies and visualize in oscilloscope. * By changing the signal generator output frequency, we could practically see band pass filter, filtering frequencies outside the range specified and visualize in oscilloscope. |
| 8 | 7/05/19  (Tuesday) | 10:00am  to 05:00pm | 6 | COMMUNICATION  Two types of communication available. Wired and Wireless communication. In Wired communication, there are two further classifications. 1) Serial Communication. 2) Parallel communication. Parallel is complex to implement always, because of additional communication lines. Serial communication solves the problem with fewer lines for communication. Eg. Internal Memory and processor communication are an example of parallel communication. Peripherals (Mouse, Keyboard) and processor is an example of serial communication. Wirless is always a serial communication.  All serial transmitters are parallel in, serial out shift registers. All serial receivers are serial in, parallel out shift registers.  In serial communication, two type of synchronization. 1) Clock based synchronization (sync) 2) Baud rate based synchronization (async). Clock based communication is typically used for short distances (e.g on board communication). Baud rate based communication is used for long range distances (e.g between boards, between different locations).  E.g Baud rates are 1200, 2400, 4800, 9600 11500 etc. High transmission, low receiving speed have problem. The solution is to agreed upon the speed of the transmission (baud rate) prior to the communication.  Every falling edge of the clock it samples for the data and if its find high, that is considered as 1 and if its low its considered as 0. Attenuation happens in clock/data signal when there is long range signal.  Clock based communication can be of multiple types I2C [SDA, SCL], SPI [MOSI, MISO, SS, SCLK]. Clock based communication doesn’t require connectors or external cables.  Baud rate communication has multiple types UART[TxD,RxD], USB [D+, D-], CAN [CAN+, CAN-], Ethernet [TX+ Tx-, Rx+ Rx-], MODBus.  In 8051, TI flag is set when the data has completely transmitted, similarly RI flag is set when data is completely received. The data to be transmitted or received in SBUF register.  **UART-**  1.Time interval between first byte and second byte is predictable in synchronous communication  Eg. Downloading an image  2. Time interval between first byte and second byte is not predictable in Asynchronous communication  Eg.Data transfer from keyboard to CPU in desktop  **ASYNCHRONOUS COMMUNICATION**  Transition from high to low in the receiver pin will act as start bit and it allows the bytes which follows the start bit till a stop bit is received which is low to high  Serial interrupt’s Interrupt service routine location is 0x002B   * Serial communication for ascii characters between two 8051 microcontrollers were tries. * Entry/Exit system with seven segment display using serial communication between two boards were tried. |