

**Introduction to Embedded System Design**  
**Professor Dhananjay V. Gadre**  
**Netaji Subhas University of Technology, New Delhi**  
**Lecture 8**  
**Elements of Microcontroller Ecosystem Continued**

(Refer Slide Time: 00:35)

**Selecting a Suitable Embedded Controller**

- Peripheral Features
- Memory
- Packaging ←
- Grade (Commercial, Industrial, Automotive, Military)
- Price
- Availability and lead time

But most important!

The slide features a dark background with a light-colored curved shape on the right side. A small video inset in the bottom right corner shows Professor Dhananjay V. Gadre sitting at a desk with a laptop, with a presentation slide visible behind him that includes the text 'MSP430'.

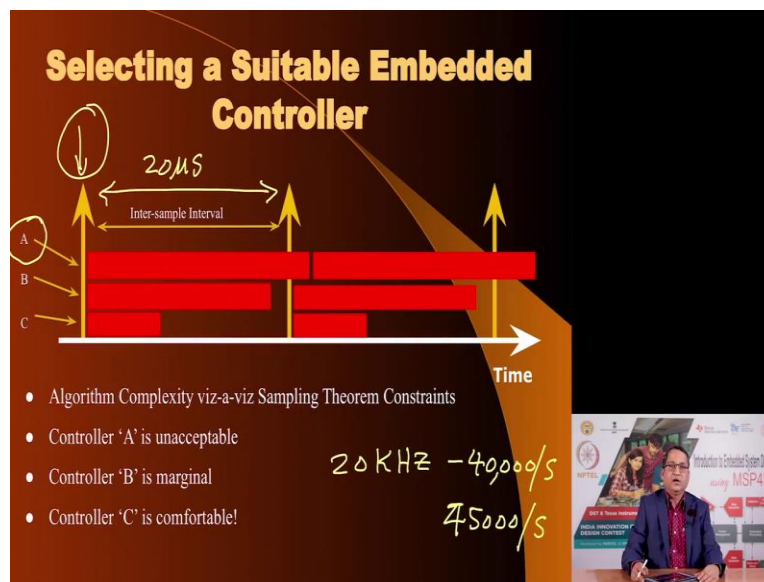
Here I have listed out certain check points that you would consider when you are selecting embedded micro controller and they would be rather your micro controller as peripheral features whether it has enough memory for your application what about the packaging because packaging would be a important consideration is it available in the package that you want because you application may not offer to much physical space.

IC are produce in many grades the commercial grade is the most common grade this is used for commodity consumer products typically the temperature of operation of such IC in the range of 0 to 70 degrees, industrial temperature range is higher than that automotive is even higher and military grade components operate at much wider temperature ranges because that is what they expect in a military application.

Then you would look at the price availability and lead time but an important component is since you are going to use micro controller in a real time application in a real time application where it is going to sample external parameters such as temperature or sound therefore it must be able to sample that signal appropriately and whether a particular micro controller is up to the job or not is determined by the maximum frequency component of that external signal.

Let me take two examples if I am looking at regulating the temperature of this room using an air conditioner. I would need to know what is the temperature of the room so I need to sample that is I need to collect the temperature sensor and digitize the value of this temperature sensor every few seconds that is enough because the room temperature is not going to vary very fast. On the other hand if I want to look at an application such as the noise cancelling headphone then the noise cancelling application has to sample external sounds the maximum frequency of the external sound is 20 kilo hertz we can't hear beyond that.

(Refer Slide Time: 02:40)



So, we must sample 20 kilo hertz at an appropriate sampling frequency and I have illustrated that here that the time between two samples we call it inter sample interval and this would vary from application to application like I mentioned for measuring temperature the inter sample interval will be very large for sampling sound frequencies if I am going to sample 20 kilo hertz signal I must sample it in keeping with the sampling theorem (03:06) and I have to sample that signal at least twice the maximum frequency component.

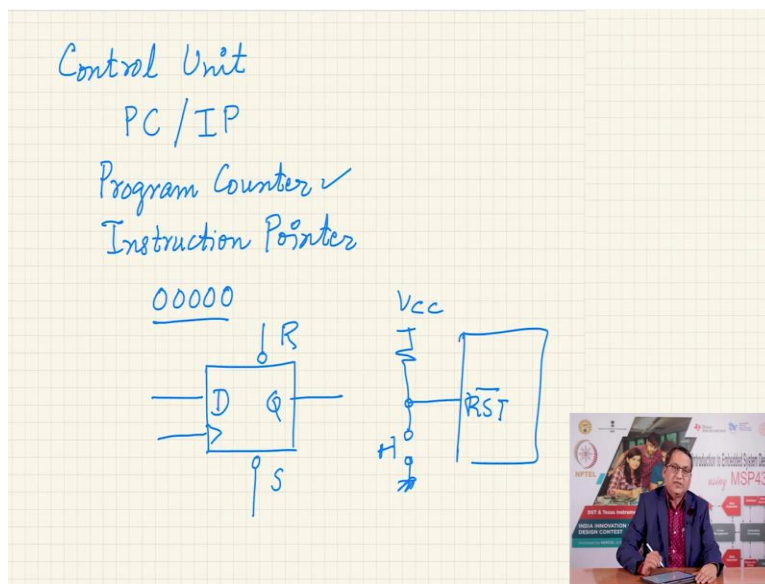
So, 20 kilo hertz signal a signal which has a maximum frequency component of 20 kilo hertz must be sampled at 40000 times per second at least and so you would sample it probably at 45000 samples per second which means the time between two samples is going to be less than I would say this is about 20 micro seconds is that right? Yes.

Therefore, what would happen in this 20 micro seconds? At this point the micro controller receives one sample. After 20 micro seconds it is going to receive one more sample. It must process the current sample that it has received here and produced an output before the next sample is due. Now we have taken three cases. Let us say we choose micro controller of type A, if the program that runs and processes this sample it would and if we take more than 20 micro seconds to complete this task it is not suitable for this application.

How can we make a usable in this situation may be we can increase the clock frequency so that the resultant time is B such that it is less than the inter sample interval than it may be a marginally acceptable solution, if we can increase the clock frequency even more that is the internal clock frequency of operation on this micro controller even more such that it takes less than half the time then it's a very comfortable situation.

In case you are not able to bring down the processing time to less than the inter sample interval, you may be forced to replace this micro controller which offers you higher performance and before you that you would consider can I increase the clock frequency operation so as to manage the completion time of the sample before the next sample is due and so therefore the value of the clock frequency is very important when you are looking at physical signals which had to follow the micro state.

(Refer Slide Time: 06:02)



Let us look at the next aspect of the ecosystem which is the reset. Micro controller requires a reset signal, why does it require a reset signal well the answer is very simple and obvious that a micro controller uses as CPU a CPU has a control unit a control unit utilizes flip flops and the values of this flip flops they are important when you apply power to the micro controller for the first time you want this flip flops to start with the known state.

Similarly, apart from the control unit you have a important resister in your micro controller or micro processor which is the program counter also sometimes called instruction pointer, let me write them here program counter or instruction pointer, this is nothing but a few flip flops arrange together to provide the initial address when the micro controller is reset.

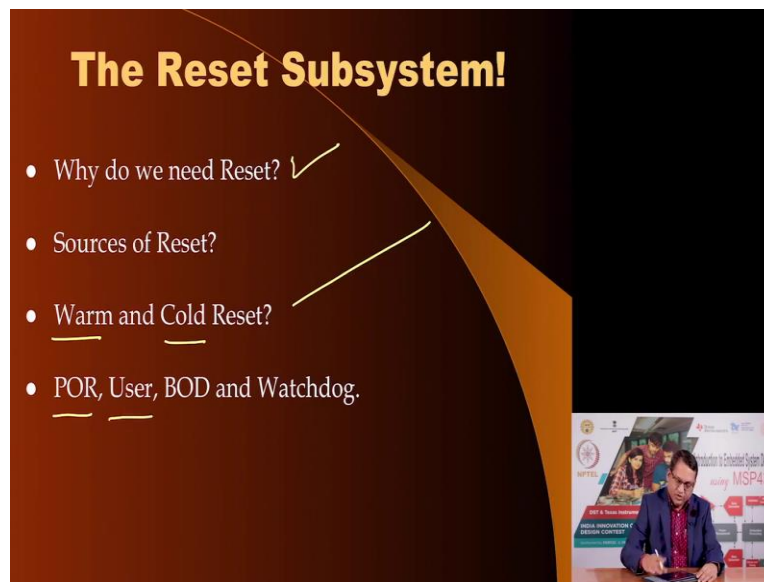
Now, if the value of this flip flops take arbitrary values that would mean that after reset the micro controller at some point may go to a certain memory location to fetch the first instruction at some other time it may go to different location and if that happens you will never be able to execute your program properly. All micro controllers require that after reset a fixed location unknown location is accessed to access the first instruction stored in the memory.

In many cases that addresses 0000 say another 0 if you like this address, so this address must be output on the address lines connected to the program memory if you do not reset the initialized the flip flops that make the program counter or instruction pointer then it may emit arbitrary values on the address first and it would be a very chaotic situation.

So, to ensure each time every time power is applied to micro processor or the micro controller, its starts with a known state internal state of the computer, it is very important that you reset, why? Because we know that a flip flop are represented with the signals say D type flip flop would have D input Q output, a clock signal this is the very minimum signals that you require but it will also offer you reset which when asserted will reset the value of Q to 0 it may also offer a signal called set if you assert set that will make Q equal to 1.

And so when you use such flip flops to create various registers the reset or set signals are utilize together by the control unit so that they can be initialized with a known value and therefore this act that when the micro controller is applied power for the first time the micro controller start with a known state this is called the resetting the microcontroller. So, we see why need a reset so that our system is going to start from the known state each time every time.

(Refer Slide Time: 09:25)



Now, there are many sources of reset, one of them as I mentioned the term every time you apply the power to the microcontroller or micro processor it should reset itself without having to be prompted and such a source of resource is called user reset. We will see other sources of reset but before that I have classified the resets to be of two types, warm reset and cold reset.

A warm reset means that the system power has not been disabled it has not been switched off and you would still like to reset the system without turning the power off. Other source of reset would be a cold reset meaning you turn the power off and when you turn it on again it would be classified as a cold reset, of all the resets that are available and here is the list of those resets you have power on reset which I have discuss that is a reset signal internally which is generated when you turn power to a micro controller system that is called power on reset.

(Refer Slide Time: 10:34)

# User Reset

Device Pinout, MSP430G2x13 and MSP430G2x53, 28-Pin Devices, TSSOP

P1.0/TA0.K/KC2/KAC2	1	DVCC	1	P2.0/DVSS	20																																																		
P1.1/TA0.DUCAR/DUCASOM/A1CA1	2	P1.2/TA0.DUCAR/DUCASOM/A1CA2	3	P1.3/TA0.DUCAR/DUCASOM/A1CA3	4	P1.4/TA0.DUCAR/DUCASOM/A1CA4	5	P1.5/TA0.DUCAR/DUCASOM/A1CA5	6	P1.6/TA0.DUCAR/DUCASOM/A1CA6	7	P1.7/TA0.DUCAR/DUCASOM/A1CA7	8	P1.8/TA0.DUCAR/DUCASOM/A1CA8	9	P1.9/TA0.DUCAR/DUCASOM/A1CA9	10	P1.10/TA0.DUCAR/DUCASOM/A1CA10	11	P1.11/TA0.DUCAR/DUCASOM/A1CA11	12	P1.12/TA0.DUCAR/DUCASOM/A1CA12	13	P1.13/TA0.DUCAR/DUCASOM/A1CA13	14	P1.14/TA0.DUCAR/DUCASOM/A1CA14	15	P1.15/TA0.DUCAR/DUCASOM/A1CA15	16	P1.16/TA0.DUCAR/DUCASOM/A1CA16	17	P1.17/TA0.DUCAR/DUCASOM/A1CA17	18	P1.18/TA0.DUCAR/DUCASOM/A1CA18	19	P1.19/TA0.DUCAR/DUCASOM/A1CA19	21	P1.20/TA0.DUCAR/DUCASOM/A1CA20	22	P1.21/TA0.DUCAR/DUCASOM/A1CA21	23	P1.22/TA0.DUCAR/DUCASOM/A1CA22	24	P1.23/TA0.DUCAR/DUCASOM/A1CA23	25	P1.24/TA0.DUCAR/DUCASOM/A1CA24	26	P1.25/TA0.DUCAR/DUCASOM/A1CA25	27	P1.26/TA0.DUCAR/DUCASOM/A1CA26	28	P1.27/TA0.DUCAR/DUCASOM/A1CA27	29	P1.28/TA0.DUCAR/DUCASOM/A1CA28	30

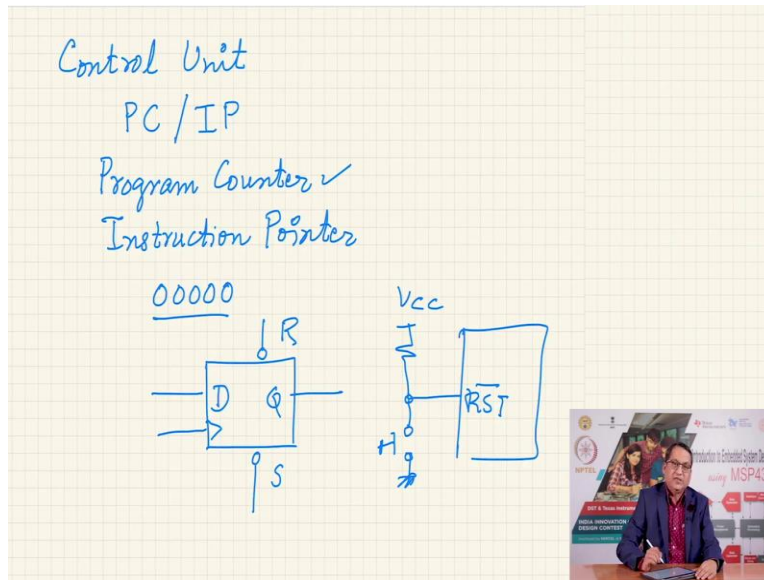
NOTE: ADC10 is available on MSP430G2x53 devices only.

(PCINT14/RESET) P0	1	(PCINT16/RXCP) P0	2	(PCINT17/TXD) P0	3	(PCINT18/INT0) P0	4	(PCINT19/CSB/INT1) P0	5	(PCINT20/CLKTR) P0	6	VCC	7	GND	8	(PCINT21/AL1/TOSC1) P0	9	(PCINT21/AL2/TOSC2) P0	10	(PCINT21/IOCB/F1) P0	11	(PCINT22/OC2A/INT0) P0	12	(PCINT23/AIN1) P0	13	(PCINT0/CLK/CP1) P0	14	PC0 (ADC5/SCL/PCINT13)	28	PC4 (ADC4/SDA/PCINT12)	27	PC3 (ADC3/PCINT11)	26	PC2 (ADC2/PCINT10)	25	PC1 (ADC1/PCINT9)	24	PC0 (ADC0/PCINT8)	23	GND	22	AREF	21	AICC	20	PB5 (SCK/PCINT5)	19	PB4 (MISO/PCINT4)	18	PB3 (MOSI/OC2A/PCINT3)	17	PB2 (SS/OC1B/PCINT2)	16	PB1 (OC1A/PCINT1)	15
--------------------	---	-------------------	---	------------------	---	-------------------	---	-----------------------	---	--------------------	---	-----	---	-----	---	------------------------	---	------------------------	----	----------------------	----	------------------------	----	-------------------	----	---------------------	----	------------------------	----	------------------------	----	--------------------	----	--------------------	----	-------------------	----	-------------------	----	-----	----	------	----	------	----	------------------	----	-------------------	----	------------------------	----	----------------------	----	-------------------	----

Another reset source is the user reset source and let me show you in the next slide here if you look at the pin out of these devices you will notice that on pin number if you see here this is the reset pin of this AVR microcontroller and on this MSP 430 microcontroller you have reset signal somewhere here.

And these pins which are designated as RST bar or reset on this two micro controller refer to the user reset meaning you would user can assert a logic level on these pins and it would reset the system even if the power system was not taken off even if the power supply was applied and usually the way to do that is you would on these pins RST bar you would connect switch.

(Refer Slide Time: 11:30)



So, every time you want to assert a user reset the user simply presses this switch, it pulls the reset pin to ground this reset the internal circuitry and when the switch is released it would allow the circuit to reset and then it will start performing its expected task.

(Refer Slide Time: 12:00)

### The Reset Subsystem!

- Why do we need Reset? ✓
- Sources of Reset?
- Warm and Cold Reset?
- POR, User, BOD and Watchdog.

Brown out detector

V<sub>CC</sub>  
T

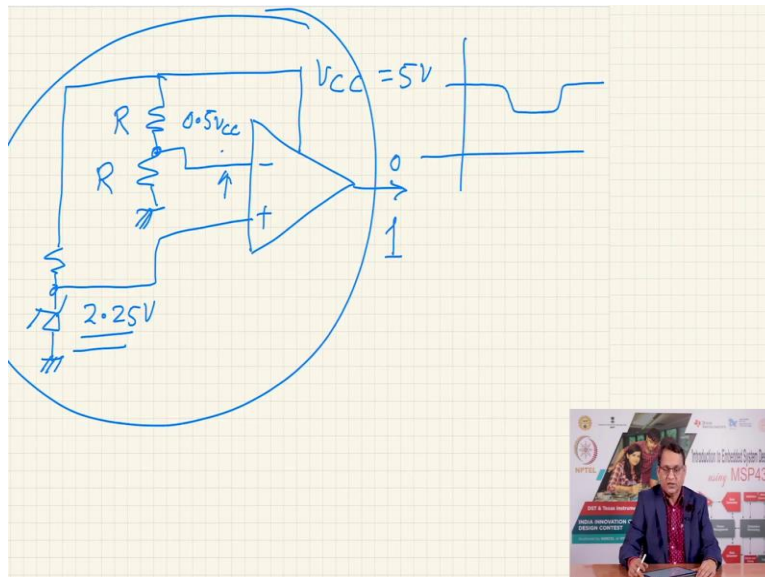
The third type of reset is called BOD, BOD stands for Brown Out Detector, so what is the brown out condition because you have a detector which detect the Brown Out Condition, brown out condition is that if this is my as a function of time this is my V<sub>CC</sub> that is the supply voltage, If at any time it deeps below certain level and for the entire duration of time it remains below that



threshold again rising to expected levels if it has fallen below this threshold here then this may be detrimental to this operation of the processor.

It may corrupt some register values it may write wrong value into flash memory or EEPROM if the controller was performing some right operations and it may be better to reset the system rather than go ahead with this corrupting activities and this is where a Brown Out Detector plays a role if this Brown Out Detector has been enabled then it would generate an internal reset which will hold the internal circuit in microcontroller in a perpetual state of reset till the power supply voltage level resumes and exceeds the minimum threshold that is required.

(Refer Slide Time: 13:37)



Let me explain the operation in using an external circuit, suppose you had you take a comparator and I would have minus and plus signals of a comparator it would have an output here what I will do is it is also power with a power supplier say VCC which is a power supplier of a microcontroller, I am going to take a voltage divider made out of two equal value resistor and therefore the voltage here would be 0.5 VCC if both this resistors are of equal values.

Let us say that VCC is 5 volts therefore the voltage you expect at this point when the supply voltage is 5 volts will be 2.5 volts. I take another little circuit using a resistor and a reference voltage diode such as a Zener, I connect it to the positive input and let say this is 2.25 volts reference diode.



Now, as long as the power supply is 5 volts the input is going to be 2.5 volts and therefore the output here is going to be 0. But at any point of time if this power supply instead of being steady at 5 volts instead of that if it falls to a low value such that the voltage here becomes less than 2.25 volts the output here will go to 1 and this signals could be use to reset the internal reset circuitry of the microcontroller and hold it in the reset till such time the voltage resumes back to the normal values or at least exceeds above the threshold at which the brown out detector kicks in.

So, this one can use one can think of such a circuit that has been integrated inside the microcontroller but it functions in this way that I have illustrated using (())(15:31) and couple of resister and a reference diode. So, this is how a brown out detector works internally and one has to be aware that their microcontroller offers such features and in case they expect the power supply to be you know show such behavior where the power supply voltage is fluctuating then they should enable the brown out detector so that their entire system keeps working properly.

(Refer Slide Time: 16:12)

## The Reset Subsystem!

- Why do we need Reset? ✓
- Sources of Reset?
- Warm and Cold Reset?
- POR, User, BOD and Watchdog.

Brown out detector

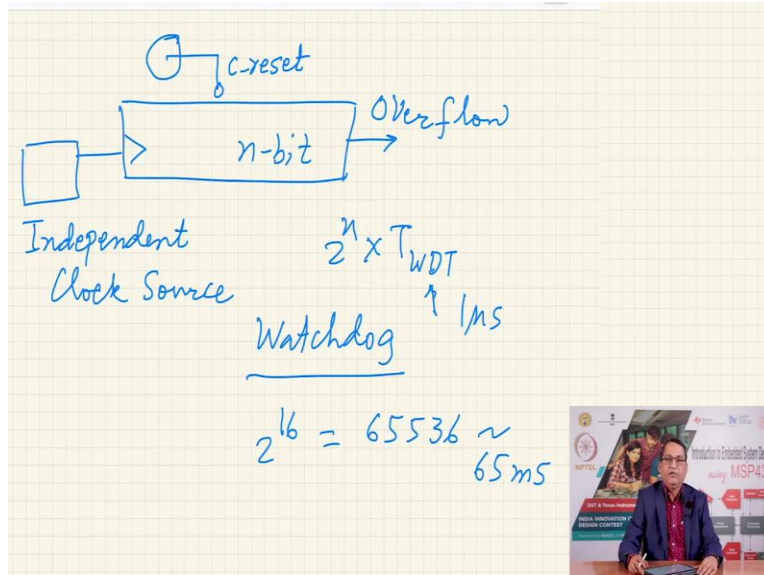
VCC

T

The fourth type of reset source is this what is called as watchdog timer and watchdog timer is common features these days in microcontroller as a name suggest it is like a watchdog at home which would start barking if it finds anything other than usual intruder or something like that in a microcontroller system if the program gets stuck for a whatever reason the watchdog timer could be used to redeem the system to comeback out of the stuck situation and start operation again

and simple way would be to reset it. Unfortunately, if the system is out of human reach where it cannot be manually reset using the user pin then the only way would be watchdog reset and let me straight again with a small circuit diagram how it works.

(Refer Slide Time: 17:03)




So, imagine that you have a counter and this counter would have can be fed with the clock source and let say it is an independent source of clock and let say this is a n bit counter, so it would count 2 raise to power n clock pulses at the end of which overflow pin of the counter overflow let me write it completely pin would become one and if this would becomes one it can be use to reset the system.

Now, obviously you do not want this timer, this is your watchdog timer you do not want this watchdog timer to overflow, so how do you ensure that it does not overflow because that overflow situation is going to reset the system that any counter can also have its own reset we let us call it C reset that is counter reset and this pin could be connected is actually part of the resistor and the microcontroller program would periodically so if I know that 2 raise to power n into the clock frequency here T I will use a name so there is not confusing with the clock frequency of the system WDT watchdog timer frequency.

Let us say this is 1 mega hertz clock so therefore it is a one micro second time period, let say n 16 therefore it is 2 raise to power 16 clock pulses required to overflow this, this is about 65536 clock pulses. So, roughly this will gives you a 65 mille seconds period within which if you can

reset the timer it will not reset you, if you forget to reset it for whatever reason that imagine that the program has suck and cannot reset this timer then the timer will overflow and it will then reset the system bringing back sanity into the system this is how a watchdog timer reset works. So, we have seen at all the various ways resetting the microcontroller now let us move forward, this I have already illustrated the user reset pins on two popular microcontrollers.

(Refer Slide Time: 19:43)

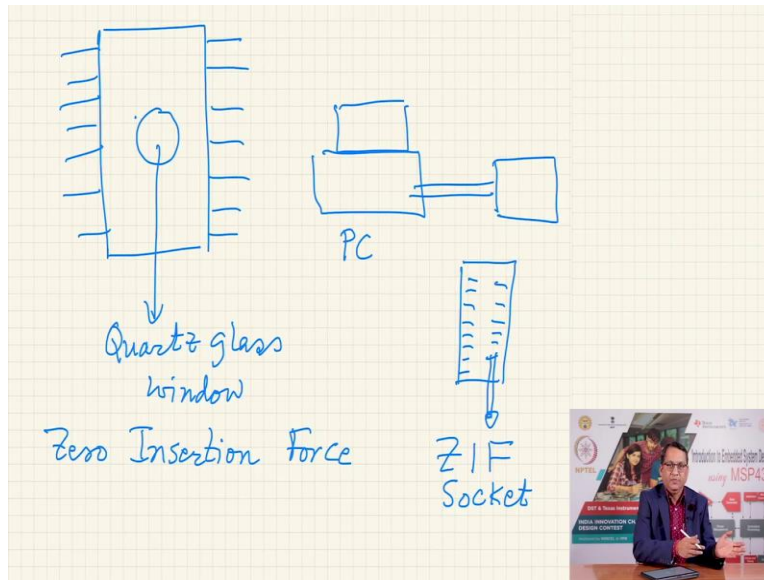


## Program Download Capability

- Older Methods (EPROM Eraser + Universal Device Programmer).
- Evolution of Flash Memory → ISP.
- Most Popular using SPI.
- JTAG: Dual use. Testing as well as Program Download
- SWD
- IAP using Bootloader.
- UPM + BPM.

The other aspect that you require as part of the microcontroller ecosystem is the ability to download program from the development setup in to the memory of the microcontroller and I am going to go through little bit of history here in the past one of the popular method was to use EPROM to store the program memory and to erase the EPROM you require a source of ultra violet light, so to erase it you would have a box which ha UV lamp with the lid you put your EPROM inside it, it would have a timer to say 15 to 20 minutes of dosage of the UV light required to erase the contents of the EPROM, EPROM typically would have a window.

(Refer Slide Time: 20:39)



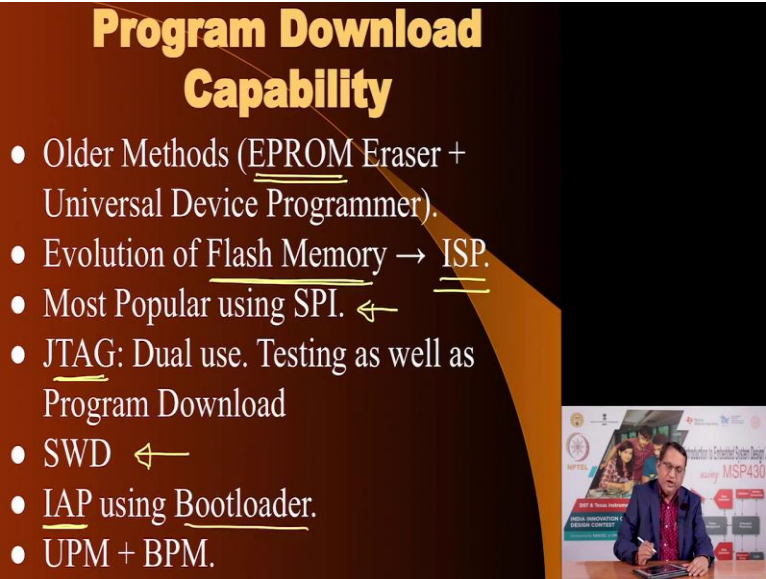
So, if this is your IC and this are the pins I am just drawing it for the sake of reference you would actually have a window this is a quartz glass window, why quartz why not normal glass? Because normal glass absorbs UV light and so these windows were made of quartz glass which is expensive type of glass, which is why these IC's were also very expensive but there were no other option and so they would use a EPROM to first erase it and then put it in a special system special programmer connected to your PC through some protocols, on this programmer you would have socket with a lever where you raise the lever it opens up the pins you drop the EPROM into the pins of the sockets and then you press that lever.

That special socket with something like this, these where the pins of the socket this was the special socket this is still in use in test equipments is called ZIF socket and it stands zero insertion force type of socket why? Because if you put a regular socket and you were to insert an integrated circuit in an out of it ends of time or hundred times a day that socket will go bad it will not make connection appropriately with the pins and may not be able to program it and therefore you wanted (())(22:27) socket which can take insertions and removals of external IC's hundreds of times a day over several of years of its operation and therefore special socket was used ZIF socket.

So, you would have this EPROM programmer or a device programmer which such a socket you would insert your EPROM into the socket after you have erased it then you would burn the so-

called burn the program into the EPROM then you would take the EPROM out of it and put it back into your target system and test it and if it did not work again you would repeat this cycle, so obviously it was slow process of compiling and downloading program into the memory of the microcontroller.

(Refer Slide Time: 23:13)



## Program Download Capability

- Older Methods (EPROM Eraser + Universal Device Programmer).
- Evolution of Flash Memory → ISP.
- Most Popular using SPI. ←
- JTAG: Dual use. Testing as well as Program Download
- SWD ←
- IAP using Bootloader.
- UPM + BPM.

The slide features a dark background with a light-colored curved shape on the right side. A small inset image in the bottom right corner shows a man in a blue shirt sitting at a desk with a laptop, presenting. Behind him is a banner for a course titled 'Introduction to Embedded System Design using MSP430'.

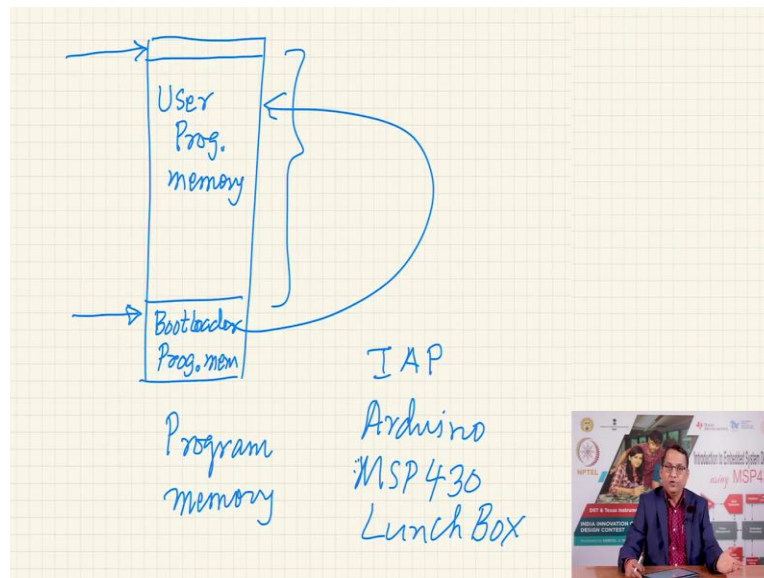
In the mid nineties there was introduction of E square a special type of E square PROM memory which was called flash and the flash memory started getting integrated into microcontrollers for the program memory that allowed user to program these memories without having to take it out of the system, without having to take the microcontroller out of the system you could program it using a feature called ISP that is in system programming.

To offer in system programming some communication interface was required and the most popular interface that is used is the SPI bus we have seen that SPI is one of the communication protocol that is available on almost all varieties of microcontroller and so this was used to off implement the ISP functionality. In more advanced microcontrollers they would have JTAG Joint Test Action Group protocol which had dual use, one of the use would be to program the internal memory and the other is would be to use it for testing the microcontroller using the on chip deep bugger that I mentioned.

Another popular protocol would be for testing would be serial (I2C)(24:32) this is a more efficient protocol which less uses less number of pins than the JTAG and another way to download

program into your code is a method code in application programming meaning you do not even stop your application from running while the system is running you can download a code into it using a technique called boot loader.

(Refer Slide Time: 25:01)



Bootloader, let me illustrate that also so the entire program memory of your system would be split in two parts, the chunk major chunk of your program, this is your program memory the major chunk of your program memory would be this one which would be called user program memory and a smaller chunk would be called bootloader program memory and when you reset the system it would normally go into the first location of the user program memory.

But if the bootloader has been enabled it would the program counter will take the program and start executing the bootloader, the bootloader would communicate using a onboard communication protocol, it would communicate and usually this communication protocol would be UART or if it as USB it would be USB and we will try to establish a contact with the host computer the host computer would have a corresponding program which would establish communication with the bootloader and then it would transfer the memory bites the program memory bites, sequentially serially one after the other the microcontroller the bootloader program would receive it and would actually write into the user program memory.

After the program transferred is complete it would transfer the control to the first memory location of the user program memory and user program execution would begin, this method is



called in application programming and using the bootloader method this is commonly used in your Arduino platforms and low and behold the MSP 430 lunch box that you all are going to be working with, also uses this bootloader method to download code from your PC into the memory of the microcontroller.

MSP 430 microcontroller come with the default bootloader program loaded into the memory at manufacture times so we do not have to worry about having to write or load this program into the bootloader program memory area and so it is a good to go system MSP 430 as long as invoking the in application programming feature of MSP 430 and we are going to use that.

This complete the program download capability or rather the feature of microcontrollers and you as designer must select one or the other option so that you can download code from the desktop computer into the memory so that your system can bug the development of the system can continue unhindered.

(Refer Slide Time: 28:25)

**Power Supply for microcontrollers**

Device Pinout, MSP430G2x13 and MSP430G2x53, 28-Pin Devices, TSSOP

P1.0TAL0UCAR0UCAG080M1A1CA1	1	20	PC5 (ADCS0LPCNT13)
P1.1TAL0UCAR0UCAG080M1A1CA1	2	21	PC4 (ADCS0APCNT12)
P1.2TAL0UCAR0UCAG080M1A1CA1	3	22	PC3 (ADCS0PCNT11)
P1.3TAL0UCAR0UCAG080M1A1CA1	4	23	PC2 (ADCS0PCNT10)
P1.4BNC0UCB8TEUCAC0UCREF+AREF+AREF+AREF+AREF+AREF	5	24	PC1 (ADCS0PCNT9)
P1.5TAL0UCAR0UCAG080M1A1CA1	6	25	PC0 (ADCS0PCNT8)
P1.6TAL0UCAR0UCAG080M1A1CA1	7	26	PC7 (ADCS7PCNT7)
P1.7TAL0UCAR0UCAG080M1A1CA1	8	27	PC6 (ADCS6PCNT6)
P1.8TAL0UCAR0UCAG080M1A1CA1	9	28	PC5 (ADCS5PCNT5)
P1.9TAL0UCAR0UCAG080M1A1CA1	10	29	PC4 (ADCS4PCNT4)
P1.10TAL0UCAR0UCAG080M1A1CA1	11	30	PC3 (ADCS3PCNT3)
P1.11TAL0UCAR0UCAG080M1A1CA1	12	31	PC2 (ADCS2PCNT2)
P1.12TAL0UCAR0UCAG080M1A1CA1	13	32	PC1 (ADCS1PCNT1)
P1.13TAL0UCAR0UCAG080M1A1CA1	14	33	PC0 (ADCS0PCNT0)
P1.14TAL0UCAR0UCAG080M1A1CA1	15	34	PC7 (ADCS7PCNT7)
P1.15TAL0UCAR0UCAG080M1A1CA1	16	35	PC6 (ADCS6PCNT6)
P1.16TAL0UCAR0UCAG080M1A1CA1	17	36	PC5 (ADCS5PCNT5)
P1.17TAL0UCAR0UCAG080M1A1CA1	18	37	PC4 (ADCS4PCNT4)
P1.18TAL0UCAR0UCAG080M1A1CA1	19	38	PC3 (ADCS3PCNT3)
P1.19TAL0UCAR0UCAG080M1A1CA1	20	39	PC2 (ADCS2PCNT2)
P1.20TAL0UCAR0UCAG080M1A1CA1	21	40	PC1 (ADCS1PCNT1)
P1.21TAL0UCAR0UCAG080M1A1CA1	22	41	PC0 (ADCS0PCNT0)
P1.22TAL0UCAR0UCAG080M1A1CA1	23	42	PC7 (ADCS7PCNT7)
P1.23TAL0UCAR0UCAG080M1A1CA1	24	43	PC6 (ADCS6PCNT6)
P1.24TAL0UCAR0UCAG080M1A1CA1	25	44	PC5 (ADCS5PCNT5)
P1.25TAL0UCAR0UCAG080M1A1CA1	26	45	PC4 (ADCS4PCNT4)
P1.26TAL0UCAR0UCAG080M1A1CA1	27	46	PC3 (ADCS3PCNT3)
P1.27TAL0UCAR0UCAG080M1A1CA1	28	47	PC2 (ADCS2PCNT2)
P1.28TAL0UCAR0UCAG080M1A1CA1	29	48	PC1 (ADCS1PCNT1)
P1.29TAL0UCAR0UCAG080M1A1CA1	30	49	PC0 (ADCS0PCNT0)
P1.30TAL0UCAR0UCAG080M1A1CA1	31	50	PC7 (ADCS7PCNT7)
P1.31TAL0UCAR0UCAG080M1A1CA1	32	51	PC6 (ADCS6PCNT6)
P1.32TAL0UCAR0UCAG080M1A1CA1	33	52	PC5 (ADCS5PCNT5)
P1.33TAL0UCAR0UCAG080M1A1CA1	34	53	PC4 (ADCS4PCNT4)
P1.34TAL0UCAR0UCAG080M1A1CA1	35	54	PC3 (ADCS3PCNT3)
P1.35TAL0UCAR0UCAG080M1A1CA1	36	55	PC2 (ADCS2PCNT2)
P1.36TAL0UCAR0UCAG080M1A1CA1	37	56	PC1 (ADCS1PCNT1)
P1.37TAL0UCAR0UCAG080M1A1CA1	38	57	PC0 (ADCS0PCNT0)
P1.38TAL0UCAR0UCAG080M1A1CA1	39	58	PC7 (ADCS7PCNT7)
P1.39TAL0UCAR0UCAG080M1A1CA1	40	59	PC6 (ADCS6PCNT6)
P1.40TAL0UCAR0UCAG080M1A1CA1	41	60	PC5 (ADCS5PCNT5)
P1.41TAL0UCAR0UCAG080M1A1CA1	42	61	PC4 (ADCS4PCNT4)
P1.42TAL0UCAR0UCAG080M1A1CA1	43	62	PC3 (ADCS3PCNT3)
P1.43TAL0UCAR0UCAG080M1A1CA1	44	63	PC2 (ADCS2PCNT2)
P1.44TAL0UCAR0UCAG080M1A1CA1	45	64	PC1 (ADCS1PCNT1)
P1.45TAL0UCAR0UCAG080M1A1CA1	46	65	PC0 (ADCS0PCNT0)
P1.46TAL0UCAR0UCAG080M1A1CA1	47	66	PC7 (ADCS7PCNT7)
P1.47TAL0UCAR0UCAG080M1A1CA1	48	67	PC6 (ADCS6PCNT6)
P1.48TAL0UCAR0UCAG080M1A1CA1	49	68	PC5 (ADCS5PCNT5)
P1.49TAL0UCAR0UCAG080M1A1CA1	50	69	PC4 (ADCS4PCNT4)
P1.50TAL0UCAR0UCAG080M1A1CA1	51	70	PC3 (ADCS3PCNT3)
P1.51TAL0UCAR0UCAG080M1A1CA1	52	71	PC2 (ADCS2PCNT2)
P1.52TAL0UCAR0UCAG080M1A1CA1	53	72	PC1 (ADCS1PCNT1)
P1.53TAL0UCAR0UCAG080M1A1CA1	54	73	PC0 (ADCS0PCNT0)
P1.54TAL0UCAR0UCAG080M1A1CA1	55	74	PC7 (ADCS7PCNT7)
P1.55TAL0UCAR0UCAG080M1A1CA1	56	75	PC6 (ADCS6PCNT6)
P1.56TAL0UCAR0UCAG080M1A1CA1	57	76	PC5 (ADCS5PCNT5)
P1.57TAL0UCAR0UCAG080M1A1CA1	58	77	PC4 (ADCS4PCNT4)
P1.58TAL0UCAR0UCAG080M1A1CA1	59	78	PC3 (ADCS3PCNT3)
P1.59TAL0UCAR0UCAG080M1A1CA1	60	79	PC2 (ADCS2PCNT2)
P1.60TAL0UCAR0UCAG080M1A1CA1	61	80	PC1 (ADCS1PCNT1)
P1.61TAL0UCAR0UCAG080M1A1CA1	62	81	PC0 (ADCS0PCNT0)
P1.62TAL0UCAR0UCAG080M1A1CA1	63	82	PC7 (ADCS7PCNT7)
P1.63TAL0UCAR0UCAG080M1A1CA1	64	83	PC6 (ADCS6PCNT6)
P1.64TAL0UCAR0UCAG080M1A1CA1	65	84	PC5 (ADCS5PCNT5)
P1.65TAL0UCAR0UCAG080M1A1CA1	66	85	PC4 (ADCS4PCNT4)
P1.66TAL0UCAR0UCAG080M1A1CA1	67	86	PC3 (ADCS3PCNT3)
P1.67TAL0UCAR0UCAG080M1A1CA1	68	87	PC2 (ADCS2PCNT2)
P1.68TAL0UCAR0UCAG080M1A1CA1	69	88	PC1 (ADCS1PCNT1)
P1.69TAL0UCAR0UCAG080M1A1CA1	70	89	PC0 (ADCS0PCNT0)
P1.70TAL0UCAR0UCAG080M1A1CA1	71	90	PC7 (ADCS7PCNT7)
P1.71TAL0UCAR0UCAG080M1A1CA1	72	91	PC6 (ADCS6PCNT6)
P1.72TAL0UCAR0UCAG080M1A1CA1	73	92	PC5 (ADCS5PCNT5)
P1.73TAL0UCAR0UCAG080M1A1CA1	74	93	PC4 (ADCS4PCNT4)
P1.74TAL0UCAR0UCAG080M1A1CA1	75	94	PC3 (ADCS3PCNT3)
P1.75TAL0UCAR0UCAG080M1A1CA1	76	95	PC2 (ADCS2PCNT2)
P1.76TAL0UCAR0UCAG080M1A1CA1	77	96	PC1 (ADCS1PCNT1)
P1.77TAL0UCAR0UCAG080M1A1CA1	78	97	PC0 (ADCS0PCNT0)
P1.78TAL0UCAR0UCAG080M1A1CA1	79	98	PC7 (ADCS7PCNT7)
P1.79TAL0UCAR0UCAG080M1A1CA1	80	99	PC6 (ADCS6PCNT6)
P1.80TAL0UCAR0UCAG080M1A1CA1	81	100	PC5 (ADCS5PCNT5)
P1.81TAL0UCAR0UCAG080M1A1CA1	82	101	PC4 (ADCS4PCNT4)
P1.82TAL0UCAR0UCAG080M1A1CA1	83	102	PC3 (ADCS3PCNT3)
P1.83TAL0UCAR0UCAG080M1A1CA1	84	103	PC2 (ADCS2PCNT2)
P1.84TAL0UCAR0UCAG080M1A1CA1	85	104	PC1 (ADCS1PCNT1)
P1.85TAL0UCAR0UCAG080M1A1CA1	86	105	PC0 (ADCS0PCNT0)
P1.86TAL0UCAR0UCAG080M1A1CA1	87	106	PC7 (ADCS7PCNT7)
P1.87TAL0UCAR0UCAG080M1A1CA1	88	107	PC6 (ADCS6PCNT6)
P1.88TAL0UCAR0UCAG080M1A1CA1	89	108	PC5 (ADCS5PCNT5)
P1.89TAL0UCAR0UCAG080M1A1CA1	90	109	PC4 (ADCS4PCNT4)
P1.90TAL0UCAR0UCAG080M1A1CA1	91	110	PC3 (ADCS3PCNT3)
P1.91TAL0UCAR0UCAG080M1A1CA1	92	111	PC2 (ADCS2PCNT2)
P1.92TAL0UCAR0UCAG080M1A1CA1	93	112	PC1 (ADCS1PCNT1)
P1.93TAL0UCAR0UCAG080M1A1CA1	94	113	PC0 (ADCS0PCNT0)
P1.94TAL0UCAR0UCAG080M1A1CA1	95	114	PC7 (ADCS7PCNT7)
P1.95TAL0UCAR0UCAG080M1A1CA1	96	115	PC6 (ADCS6PCNT6)
P1.96TAL0UCAR0UCAG080M1A1CA1	97	116	PC5 (ADCS5PCNT5)
P1.97TAL0UCAR0UCAG080M1A1CA1	98	117	PC4 (ADCS4PCNT4)
P1.98TAL0UCAR0UCAG080M1A1CA1	99	118	PC3 (ADCS3PCNT3)
P1.99TAL0UCAR0UCAG080M1A1CA1	100	119	PC2 (ADCS2PCNT2)
P1.100TAL0UCAR0UCAG080M1A1CA1	101	120	PC1 (ADCS1PCNT1)
P1.101TAL0UCAR0UCAG080M1A1CA1	102	121	PC0 (ADCS0PCNT0)
P1.102TAL0UCAR0UCAG080M1A1CA1	103	122	PC7 (ADCS7PCNT7)
P1.103TAL0UCAR0UCAG080M1A1CA1	104	123	PC6 (ADCS6PCNT6)
P1.104TAL0UCAR0UCAG080M1A1CA1	105	124	PC5 (ADCS5PCNT5)
P1.105TAL0UCAR0UCAG080M1A1CA1	106	125	PC4 (ADCS4PCNT4)
P1.106TAL0UCAR0UCAG080M1A1CA1	107	126	PC3 (ADCS3PCNT3)
P1.107TAL0UCAR0UCAG080M1A1CA1	108	127	PC2 (ADCS2PCNT2)
P1.108TAL0UCAR0UCAG080M1A1CA1	109	128	PC1 (ADCS1PCNT1)
P1.109TAL0UCAR0UCAG080M1A1CA1	110	129	PC0 (ADCS0PCNT0)
P1.110TAL0UCAR0UCAG080M1A1CA1	111	130	PC7 (ADCS7PCNT7)
P1.111TAL0UCAR0UCAG080M1A1CA1	112	131	PC6 (ADCS6PCNT6)
P1.112TAL0UCAR0UCAG080M1A1CA1	113	132	PC5 (ADCS5PCNT5)
P1.113TAL0UCAR0UCAG080M1A1CA1	114	133	PC4 (ADCS4PCNT4)
P1.114TAL0UCAR0UCAG080M1A1CA1	115	134	PC3 (ADCS3PCNT3)
P1.115TAL0UCAR0UCAG080M1A1CA1	116	135	PC2 (ADCS2PCNT2)
P1.116TAL0UCAR0UCAG080M1A1CA1	117	136	PC1 (ADCS1PCNT1)
P1.117TAL0UCAR0UCAG080M1A1CA1	118	137	PC0 (ADCS0PCNT0)
P1.118TAL0UCAR0UCAG080M1A1CA1	119	138	PC7 (ADCS7PCNT7)
P1.119TAL0UCAR0UCAG080M1A1CA1	120	139	PC6 (ADCS6PCNT6)
P1.120TAL0UCAR0UCAG080M1A1CA1	121	140	PC5 (ADCS5PCNT5)
P1.121TAL0UCAR0UCAG080M1A1CA1	122	141	PC4 (ADCS4PCNT4)
P1.122TAL0UCAR0UCAG080M1A1CA1	123	142	PC3 (ADCS3PCNT3)
P1.123TAL0UCAR0UCAG080M1A1CA1	124	143	PC2 (ADCS2PCNT2)
P1.124TAL0UCAR0UCAG080M1A1CA1	125	144	PC1 (ADCS1PCNT1)
P1.125TAL0UCAR0UCAG080M1A1CA1	126	145	PC0 (ADCS0PCNT0)

NOTE: ADC10 is available on MSP430G2x53 devices only.

**AVR**

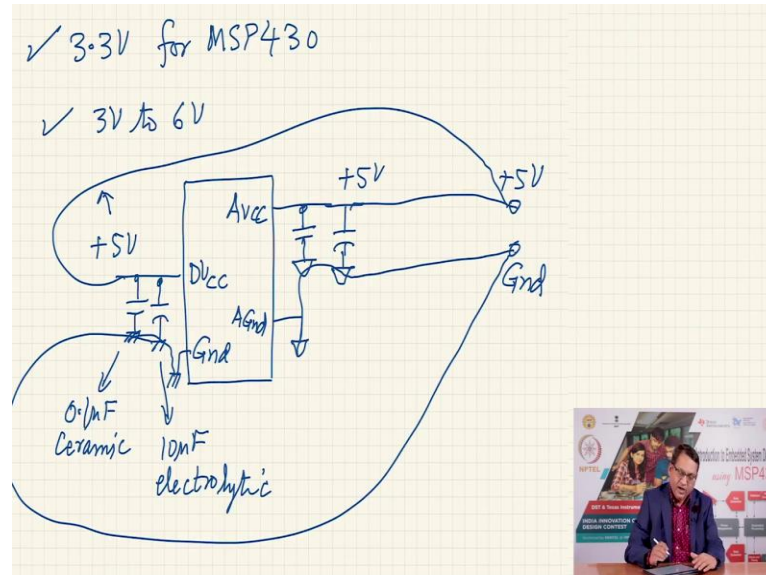
(PCNT14)RESET) P0C [ 1	26	PC5 (ADCS0LPCNT13)
(PCNT16)RXD) P0D [ 2	27	PC4 (ADCS0APCNT12)
(PCNT17)TXD) P0D [ 3	28	PC3 (ADCS0PCNT11)
(PCNT18)INT0) P0D [ 4	29	PC2 (ADCS0PCNT10)
(PCNT19)OC2BINT1) P0D [ 5	30	PC1 (ADCS0PCNT9)
(PCNT20)ICLK7) P0D [ 6	31	PC0 (ADCS0PCNT8)
(PCNT21)XTAL1) P0D [ 7	32	PC7 (ADCS7PCNT7)
(PCNT22)XTAL2) P0D [ 8	33	PC6 (ADCS6PCNT6)
(PCNT23)XTAL3) P0D [ 9	34	PC5 (ADCS5PCNT5)
(PCNT24)OC2B1) P0D [ 10	35	PC4 (ADCS4PCNT4)
(PCNT25)OC2AINT0) P0D [ 11	36	PC3 (ADCS3PCNT3)
(PCNT26)INT1) P0D [ 12	37	PC2 (ADCS2PCNT2)
(PCNT27)INT0) P0D [ 13	38	PC1 (ADCS1PCNT1)
(PCNT28)OC1CP1) P0D [ 14	39	PC0 (ADCS0PCNT0)
(PCNT29)OC1CP0) P0D [ 15	40	PC7 (ADCS7PCNT7)
(PCNT30)OC1CP1) P0D [ 16	41	PC6 (ADCS6PCNT6)
(PCNT31)OC1CP0) P0D [ 17	42	PC5 (ADCS5PCNT5)
(PCNT32)OC1CP1) P0D [ 18	43	PC4 (ADCS4PCNT4)
(PCNT33)OC1CP0) P0D [ 19	44	PC3 (ADCS3PCNT3)
(PCNT34)OC1CP1) P0D [ 20	45	PC2 (ADCS2PCNT2)
(PCNT35)OC1CP0) P0D [ 21	46	PC1 (ADCS1PCNT1)
(PCNT36)OC1CP1) P0D [ 22	47	PC0 (ADCS0PCNT0)
(PCNT37)OC1CP0) P0D [ 23	48	PC7 (ADCS7PCNT7)
(PCNT38)OC1CP1) P0D [ 24	49	PC6 (ADCS6PCNT6)
(PCNT39)OC1CP0) P0D [ 25	50	PC5 (ADCS5PCNT5)
(PCNT40)OC1CP1) P0D [ 26	51	PC4 (ADCS4PCNT4)
(PCNT41)OC1CP0) P0D [ 27	52	PC3 (ADCS3PCNT3)
(PCNT42)OC1CP1) P0D [ 28	53	PC2 (ADCS2PCNT2)
(PCNT43)OC1CP0) P0D [ 29	54	PC1 (ADCS1PCNT1)
(PCNT44)OC1CP1) P0D [ 30	55	PC0 (ADCS0PCNT0)
(PCNT45)OC1CP0) P0D [ 31	56	PC7 (ADCS7PCNT7)
(PCNT46)OC1CP1) P0D [ 32	57	PC6 (ADCS6PCNT6)
(PCNT47)OC1CP0) P0D [ 33	58	PC5 (ADCS5PCNT5)
(PCNT48)OC1CP1) P0D [ 34	59	PC4 (ADCS4PCNT4)
(PCNT49)OC1CP0) P0D [ 35	60	PC3 (ADCS3PCNT3)
(PCNT50)OC1CP1) P0D [ 36	61	PC2 (ADCS2PCNT2)
(PCNT51)OC1CP0) P0D [ 37	62	PC1 (ADCS1PCNT1)
(PCNT52)OC1CP1) P0D [ 38	63	PC0 (ADCS0PCNT0)
(PCNT53)OC1CP0) P0D [ 39	64	PC7 (ADCS7PCNT7)
(PCNT54)OC1CP1) P0D [ 40	65	PC6 (ADCS6PCNT6)
(PCNT55)OC1CP0) P0D [ 41	66	PC5 (ADCS5PCNT5)
(PCNT56)OC1CP1) P0D [ 42	67	PC4 (ADCS4PCNT4)
(PCNT57)OC1CP0) P0D [ 43	68	PC3 (ADCS3PCNT3)
(PCNT58)OC1CP1) P0D [ 44	69	PC2 (ADCS2PCNT2)
(PCNT59)OC1CP0) P0D [ 45	70	PC1 (ADCS1PCNT1)
(PCNT60)OC1CP1) P0D [ 46	71	PC0 (ADCS0PCNT0)
(PCNT61)OC1CP0) P0D [ 47	72	PC7 (ADCS7PCNT7)
(PCNT62)OC1CP1) P0D [ 48	73	PC6 (ADCS6PCNT6)
(PCNT63)OC1CP0) P0D [ 49	74	PC5 (ADCS5PCNT5)
(PCNT64)OC1CP1) P0D [ 50	75	PC4 (ADCS4PCNT4)
(PCNT65)OC1CP0) P0D [ 51	76	PC3 (ADCS3PCNT3)
(PCNT66)OC1CP1) P0D [ 52	77	PC2 (ADCS2PCNT2)
(PCNT67)OC1CP0) P0D [ 53	78	PC1 (ADCS1PCNT1)
(PCNT68)OC1CP1) P0D [ 54	79	PC0 (ADCS0PCNT0)
(PCNT69)OC1CP0) P0D [ 55	80	PC7 (ADCS7PCNT7)
(PCNT70)OC1CP1) P0D [ 56	81	PC6 (ADCS6PCNT6)
(PCNT71)OC1CP0) P0D [ 57	82	PC5 (ADCS5PCNT5)
(PCNT72)OC1CP1) P0D [ 58	83	PC4 (ADCS4PCNT4)
(PCNT73)OC1CP0) P0D [ 59	84	PC3 (ADCS3PCNT3)
(PCNT74)OC1CP1) P0D [ 60	85	PC2 (ADCS2PCNT2)
(PCNT75)OC1CP0) P0D [ 61	86	PC1 (ADCS1PCNT1)
(PCNT76)OC1CP1) P0D [ 62	87	PC0 (ADCS0PCNT0)
(PCNT77)OC1CP0) P0D [ 63	88	PC7 (ADCS7PCNT7)
(PCNT78)OC1CP1) P0D [ 64	89	PC6 (ADCS6PCNT6)
(PCNT79)OC1CP0) P0D [ 65	90	PC5 (ADCS5PCNT5)
(PCNT80)OC1CP1) P0D [ 66	91	PC4 (ADCS4PCNT4)
(PCNT81)OC1CP0) P0D [ 67	92	PC3 (ADCS3PCNT3)
(PCNT82)OC1CP1) P0D [ 68	93	PC2 (ADCS2PCNT2)
(PCNT83)OC1CP0) P0D [ 69	94	PC1 (ADCS1PCNT1)
(PCNT84)OC1CP1) P0D [ 70	95	PC0 (ADCS0PCNT0)
(PCNT85)OC1CP0) P0D [ 71	96	PC7 (ADCS7PCNT7)
(PCNT86)OC1CP1) P0D [ 72	97	PC6 (ADCS6PCNT6)
(PCNT87)OC1CP0) P0D [ 73	98	PC5 (ADCS5PCNT5)
(PCNT88)OC1CP1) P0D [ 74	99	PC4 (ADCS4PCNT4)
(PCNT89)OC1CP0) P0D [ 75	100	PC3 (ADCS3PCNT3)
(PCNT90)OC1CP1) P0D [ 76	101	PC2 (ADCS2PCNT2)
(PCNT91)OC1CP0) P0D [ 77	102	PC1 (ADCS1PCNT1)
(PCNT92)OC1CP1) P0D [ 78	103	PC0 (ADCS0PCNT0)
(PCNT93)OC1CP0) P0D [ 79	104	PC7 (ADCS7PCNT7)
(PCNT94)OC1CP1) P0D [ 80	105	PC6 (ADCS6PCNT6)
(PCNT95)OC1CP0) P0D [ 81	106	PC5 (ADCS5PCNT5)
(PCNT96)OC1CP1) P0D [ 82	107	PC4 (ADCS4PCNT4)
(PCNT97)OC1CP0) P0D [ 83	108	PC3 (ADCS3PCNT3)
(PCNT98)OC1CP1) P0D [ 84	109	PC2 (ADCS2PCNT2)
(PCNT99)OC1CP0) P0D [ 85	110	PC1 (ADCS1PCNT1)
(PCNT100)OC1CP1) P0D [ 86	111	PC0 (ADCS0PCNT0)

The last part of the ecosystem is power supply, we are going to subsequent to this we are going to have dedicated session on power supply design including that for the microcontroller but I thought I would use this opportunity to illustrate that on your microcontrollers typically you would here see two pins DVcc and DVss are two pins on MSP 430 whereas on AVR microcontroller this is the AVR at mega microcontroller, there are two sets of pin as I mentioned earlier this is the digital supply voltage and this and this is the analog supply voltage. Pins which



are available you have to refer to the data sheet as to what is the supply voltage that the microcontroller expects.

(Refer Slide Time: 29:24)



For example, MSP 430 the expected supply voltage is 3.3 volts typically whereas for AVR microcontroller it offer wider range anywhere between 3 volts to 6 volts they can operate and therefore it becomes your responsibility as a designer to provide smooth DC voltage of this or this value so that your micro controller can function properly.

When it has two sets of power supplies as in the case of the AVR, one for the digital pins digital part of the circuit inside the micro controller and the other for the unlock part of the micro controller then one has to deal with that situation with little more care if this is DVcc to represent the digital input, one very good method is first of all to decoupling capacitors two decoupling capacitors connected in parallel and the purpose is not to add the two capacitance values which is actually happens but this is typically 0.1 micro farad ceramic capacitor.

And this could be 10 micro farad electrolytic capacitor and then this is your ground and this is your ground so I ground that pin and if you have analog AVcc also you would connect a capacitor two capacitors actually and please note that I am actually using different grounding symbols to indicate that this are these do not share the same ground with the ground of the so this is I would call it AGnd and this I am you see I am grounding like this.

This could be plus 5 volt this could also be plus 5 volt but because I have connected the ground here separately the source of the pin per supply is say plus 5 volts here and this is my ground the way to ensure that these analog grounds are not directly connected to the digital grounds on the microcontroller pin is to route them separately using separate tracks similarly you have separate tracks to go here, these are all connected together and then they go here.

If you keep the track separate it would serve the purpose of isolating the noises which is often generated on the power supply pins because of the internal switching actions in the logic circuit and it would not corrupt this supply voltage. How to achieve that in real life in practical we would consider when we are looking at system design issues where we will show how to design PCB from a power a schematic, what you do to ensure that there are two parallel tracks one for providing power to the supply to the analog voltage and the other to supply voltage to the digital part of the circuit.

We will consider that but this is an important point to keep in mind in case your micro controller as separate pins for analog and digital components subsistence within your microcontroller that you design the routing you design the PCB appropriately so as to do justice to the internal components we will continue our discussion in a next lecture where we will look at the design of power supply in more detail and I will go through the various options that are available. Thank you for watching, bye bye.