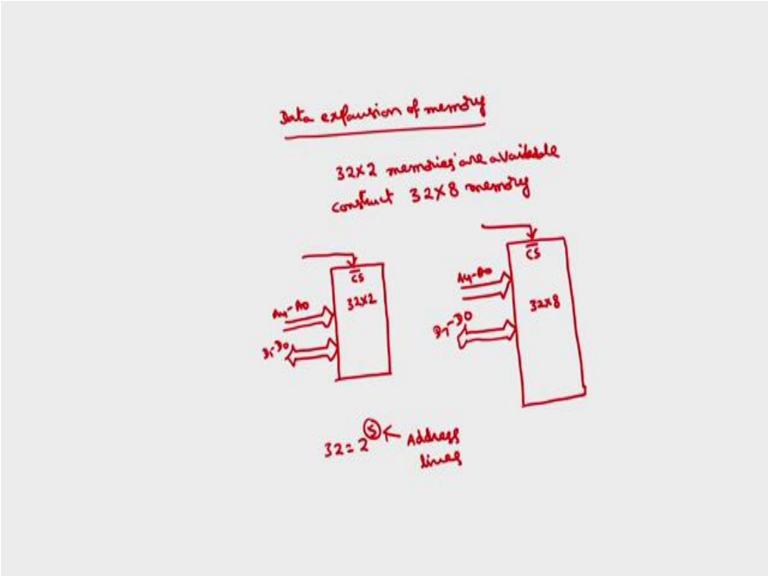
Microprocessors and Interfacing
Prof. Shaik Rafi Ahmed
Department of Electronics and Electrical Engineering
Indian Institute of Technology, Guwahati

Lecture – 16
Example I

(Refer Slide Time: 00:36)

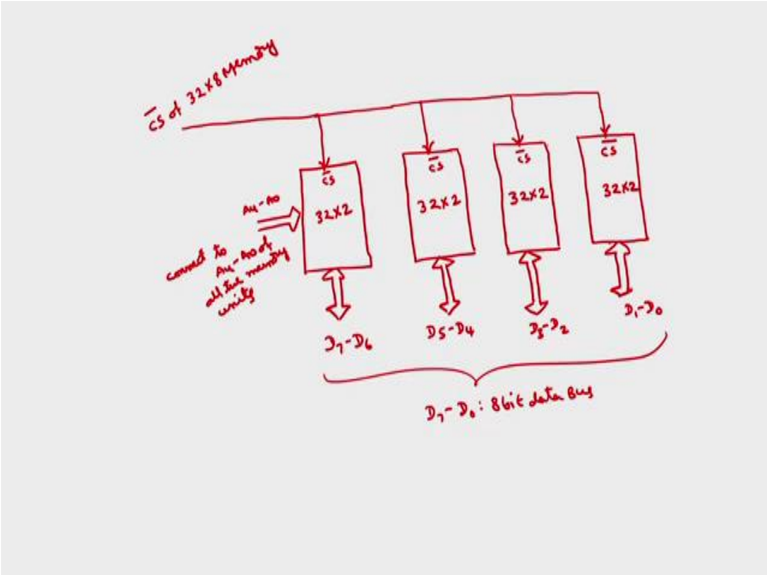


In the last class we had discussed about the address expansion of the memory. Now we will discuss about the data expansion of the memory, data expansion of memory. Suppose, you will be given 32 by 4 or 32 by 2 memories are available. If you want to construct 32 by 8 memory. Here, I am fixing this number of address lines because only I want to expand the data after that I will discuss about both expansions ok. Here 32 by 2, 32 by 2 means; we will be having 2 bit data bus and 5 bits address bus 32 is nothing, but 2 raise to the power of 5. So, 5 address lines A4 to A0.

This is 32 by 2 and two data lines D1 to D0. Of course, there is chip select and the read write bar will be there this will be have some chip select signal normal chip select will be active low as I have told and read bar write bar depends upon whether the memory is read only memory or read write memory. Now, if I take 32 by 8, the only difference is the number of address lines remain same, 32 by 8, this also will have 5 bit address lines A4 to A0 and data will be 8 bit, D7 to D0.

This also will be having some chip select signal. Now basically here is 8 by 2 is the number of 32 by 2 memory is required to construct 32 by 8, because this 32 is constant, this we have 8, we have 2, we required the 8. So, the number of 32 by 2 memories required will be simply 8 by 2 which is equal to 4, then how to construct this? For that you take four 32 by 2 memories.

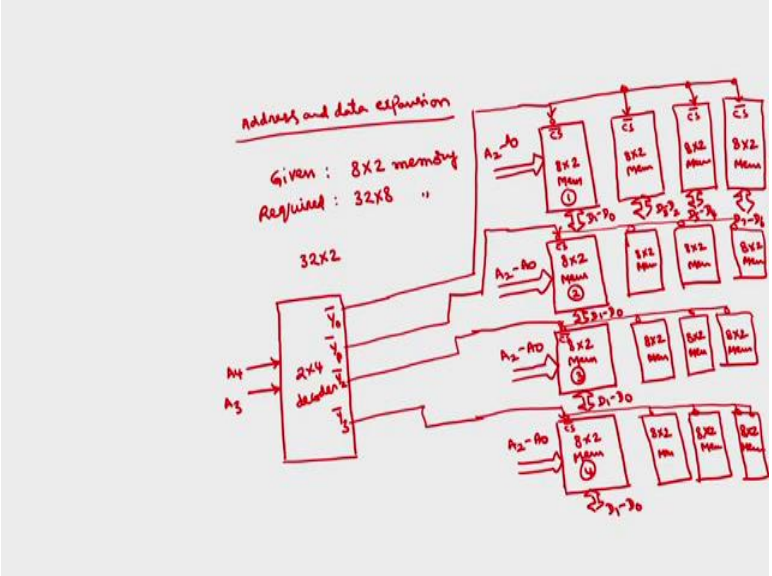
(Refer Slide Time: 03:28)



Simply connect all the chip selects together, all the 4 memories will be selected simultaneously. So, this is having same number of address lines, 5 address lines A4 to A0 you connect to connect to A4 to A0 of all the remaining memories. How to connect together, this is 32 by 2, this is 32 by 2, this is 32 by 2, this is 32 by 2, this is overall chip select signal of the required 32 by 8 memory.

So, here in this, this will give 2 bits of the data D0 to D1 or you can call as D1 to D0 MSB and LSB first you have to represent MSB and then LSB. This will give D3 to D2, this will give D5 to D4, this will give D7 to D6, total overall this will be 8 bit data bus D7 to D0. This is how we can expand the data lines. Now how to expand both address and data lines ok. Now, this overall this will acts as 32 by 8 memory.

(Refer Slide Time: 06:42)



Address and data expansion. Suppose, you are given 8 by 2 memory required is say some 32 by 8 memory. Now we have this is 8 and this is 32, address also you have to expand and this is 2 this is 8, data also you how to expand ok, you have to expand both. First, let us consider only this address expansion, hence we have discussed in the earlier class. So, in order to expand the address, so you have take the decoder. So, how many such first you construct 32 by 2, first you construct 32 by 2, later I can easily expand this to 32 by 8 ok.

To construct these 32 by 2 memory, so as you have told that we require 4 such 8 by 2 memories. This is one of 8 by 2 memory, this is number 2. You can call this as a number 1, number 2, number 3, number 4. So, each one will be having, because 8 is the number of locations, 2 raised to the power of 3 is 8. So, 3 bits of the address bus. A2 to A0, you have to connect to A2 to A0 of all the address lines and data lines will be this will give 2, this will give 2, this will give 2 and so on.

This will be having 2 bits of the data line you can give otherwise at the bottom this will give 2 bits of the data line, this will give 2 bits of the data line, this will give 2 bits of the data line, this will give 2 bits of the data line, each one will be having individual chips select signals. We can represent chip select bar, so we can have a bubble here.

Now, how to select chip selects of these signals? As we have discussed in this earlier class, you have to use 2 to 4 decoder. So, here this 32 by 8, we will be having 5 address lines, here we have 3 address lines, the remaining 2 address lines we have to use as a inputs for the decoder. 2 to 4 decoder A4, A3, this Y0 bar, I am using active low as you have told Y1 bar, Y2 bar, Y3 bar. So, in the earlier class we have also discussed about this that for the first 8 combinations this A4 A3 will be 0 0, for the next 0 1, next 1 0, next 1 1.

So, Y0 bar has to be connected to the chip select of the first 8 by 2 memory. Similarly Y1 has to be connected to the next one, Y2 has to be connected to the next and Y3 to last. So, in this way we can select 32 locations, the first one will give 8, second one will give 8, third one will give 8, fourth one will give 8. So, all together is 32 locations, but because this is 8 by 2, each I mean memory will give only 2 bits of the data ok.

This will give 2 bits of the data, we can call as D0 to D1 or D1 to D0, this will give 2 bits D1 to D0, this will give D1 to D0, this will give D1 to D0. But I want 8 bits of the data. So, I have to connect 4 such chips together. As I have told this is data expansion. So, we have to use 4 such chips. The only thing is chip select of all these 4 parallel chips has to be connected together.

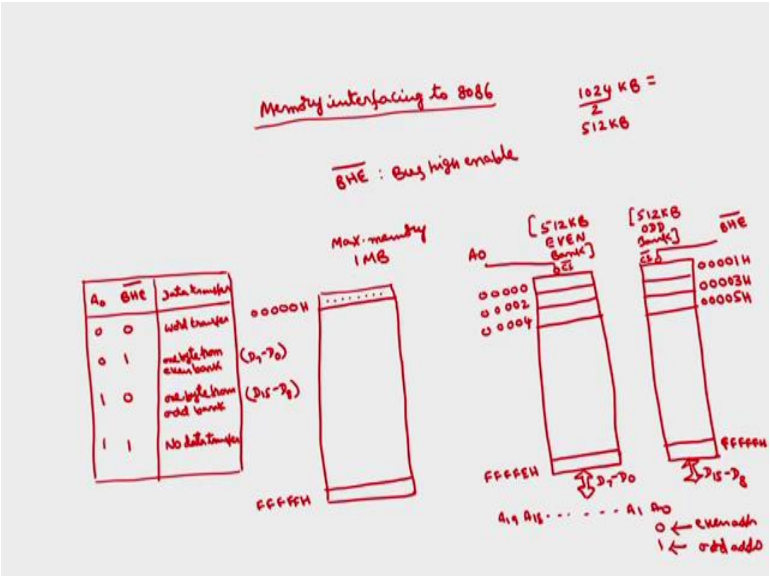
This is another 8 by 2 memory, another 8 by 2 memory we have to connect A2 to A0 of all these memories together. Only chip select is different and data is different and this will give 2 bits of the information D3 to D2 and this will give D5 to D4, this will give D7 to D6.

Similarly, here also you have to connect 4 such chips whose chip select has to be connected together, here also you will be having 4 whose chip select has to be connected together, here also 4. So, totally we need 16 such 8 by 2 memories to get 32 by 8. So, normally these type of questions will be asked in some competitive exams. So, to construct 32 by 8, how many 8 by 2 memories are required?

Here the answer is 16 8 by 2 memory, so I have to simply connect this chip select to here also, take this chip select connect together all. Similarly, here also all the chip selects you have to connect together, here also you have to connect all the chips selects together. Whereas address lines of all the A2 to A0 of all the 16 8 by 2 memories will be common, similarly read bar write bars also will be common.

This is about both address and data expansion ok. So, we need a decoder wherever the address expansion is required, decoder type of the thing and whenever you require the data expansion, similarly just you have to just connect the all the chips selects together you have to connect in parallel ok. So, with this memory concept we will go to the memory interfacing to 8086, memory interfacing to 8086.

(Refer Slide Time: 16:36)



As I have discussed in the earlier classes while discussing about this pins of 8086, there is one signal called BHE signal, BHE bar, BHE bar which is called bus high enable. So, because 8086 is a 16 bit microprocessor, we can access 16 bit of the data simultaneously. But, if I take the memory location, each memory location is capable of storing only 8 bit of the information.

So, the total maximum memory of 8086 is maximum memory of 8086 is 1 megabyte starting with 00000 H to all Fs FFFFF H, this is the maximum memory ok. But each location is capable of storing only 8 bits of information. Then the question is how to access 16 bit of the data? So, if you want to access from the 2 consecutive locations it will take 2 clock cycles.

But in one clock cycle, if I want to I mean; access 16 bit of the data because there are many operations where 16 bit data is required. So, in that case in order to get the 16 bit of the data in single clock cycle, we have to divide the memory into banks ok. Now I have to divide this into this 1 megabyte into 2 halves ok. So, 1 megabyte is nothing, but

1024 kilobytes ok. So, half of this one will be 512 kilobytes each. So, we will be having memory banks, this is even bank or odd bank 512 kilobytes, we can call this one as even bank this is 512 kilobytes odd bank.

So, all the even addresses will be present here. The first address itself is even address 0 0 0 0 0 and then the next address will be 0 0 0 0 2, 0 0 0 0 4 and so on. So, what will be last address F F F F F H will be odd address. One above this one is F F F F E H is even address. So, this will be having 512 kilobytes of even address and this will be having 512 kilobytes of odd address. What is the first odd address? 0 0 0 0 1 H, then the next one is 0 0 0 0 3 H, 0 0 0 0 5 H and so on, the last one will be F F F F F H.

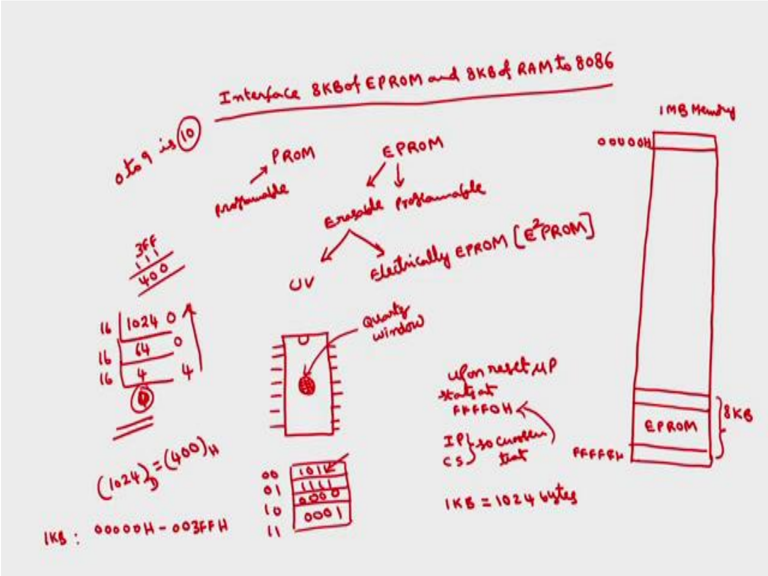
Now, we know that for even addresses A0 is 0. If I take A19 to A19 to A0, A19, A18 so on up to A1 A0 for all even addresses. This A0 is equal to 0 for even address and this is equal to 1 for odd address. So, in order to select this even bank, we are going to connect this even bank chip select bar to A0, this is chip select bar of even bank because this active low signal whenever A0 is equal to 0, then only even bank has to be selected ok, this has to be connected to A0.

Similarly, chip select bar of odd bank has to be connected to this bus high enable BHE bar. Whenever BHE bar is 0, then the odd bank will be selected whenever A0 is 0, even bank will be selected ok. Now, based on this A0 and BHE bar, we have 4 different operations. So, the form a table here A0 BHE bar. If it is 0 0, what does it mean? Both even bank and odd bank will be selected what type of data transfer will take place.

Data transfer will be word, word transfer. Either this can be read or write operation ok. Both the banks will be selected, this will give 8 bit, this will give 8 bit, total 16 bit a word. This is 01 A0 is 0 means; this is even bank BHE bar is 1 means; bus high enable is not enabled which is disabled. So, this will be only 1 byte from even bank. So, the even bank will give the data bits as this is even bank will give this data bits D7 to D0 and odd bank will give data as D15 to D8 ok.

This will be D7 to D0, the byte will be available on the data lines D7 to D0 if A0 is 0 BHE bar is 1. If A0 is 1, BHE bar is 0 then also 1 byte, but 1 byte from odd bank means; data lines are D15 to D8. If both are ones ok, no transfer occurs, no data transfer will takes place. Because, both will be disabled ok, there will no data transfer ok. So, based on this we can connect the memory to the microprocessor.

(Refer Slide Time: 24:59)



I will give one example, interface 8 kilobytes of EPROM and 8 kilobytes of RAM to 8086, this is an example. We have EPROM and RAM. So, I have discussed about this ROM which is read only memory. But what is EPROM is? ROM, there are different types of the ROMs, which is called PROM programmable read only memory, this P stands for programmable.

So, at the time of manufacturing itself, the manufacturer will store the data which we cannot change ok. So, in case of erasable programmable read only memory, EPROM means; here it is not possible to erase the data whereas, here this E stands for erasable programmable read only memory. How to erase the data? We have 2 techniques to erase the data, we can use either ultra violetically erasable or electrically erasable.

So, ultra violetically means; if I take this memory IC in the ROM, you can see in your laboratory also on electrically erasable RAM, you can find a small quartz window at the center of this IC say this is a quartz window. If I expose this window to the ultraviolet rays with the some definite wavelength for particular time, then the contents will be erased, but the drawback of this type of a ultra violetically erasable programmable read only memory is; suppose if I have some four locations each location is capable of storing 2 bits or 4 bits of information 1 0 1 1 1 1 1 1, 0 0 0 0 0 0 0 1 say.

These are the 4 addresses 0 0 0 1 1 0 1 1 are the addresses. Suppose, if want to erase only this bit I want to erase only single bit that is not possible in case of ultra violetically, the entire data will be erased that is the drawback of this one. So, to avoid that there is another technique called electrically erasable programmable read only memory. This is call electrically EPROM, electrically erasable programmable read only memory. This is also can be called as E square PROM one E stands for electrically, second E stands for erasable programmable read only memory.

Here this is possible to, I mean; erase a single bit or a single location or the entire memory. Here you can program accordingly depends upon the our requirement. So, this is how this E square PROM one of the popularly used read only memory for many of the applications. So, here I am using this EPROM this can be by default you say E square PROM, then we have random access memory. So, 8 kilo bytes of EPROM 8 kilobytes of random access memory.

So, the first step here is you have to I mean a choose appropriate memory locations, out of this 1 megabyte of the memory locations this is the total memory location is 1 megabyte starting with 0 0 0 0 0 H to F F F F F H. We can use any portion of this one for EPROM, any portion of this for RAM ok. If it is not mentioned in the statement.

Sometimes, so it will be given in the statement itself that you interface 8 kilo bytes of EPROM from say some F 0 0 0 0 H to F 1 F F F H. So, in that case you have to connect in that particular place only. If nothing is given, you can choose any of this space for EPROM and some space for RAM ok. But the usual practice is, so whenever the microprocessor is reset, then it will point to address F F F F 0 H upon reset the microprocessor starts at these address. The I P and C S will be so, chosen that so, chosen that it will go to this particular address.

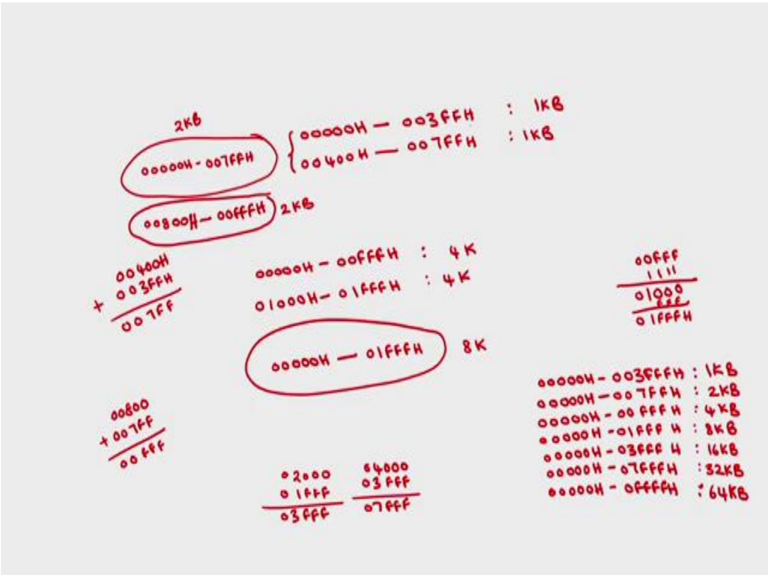
That is why normally the last portion of the memory, you have to use for EPROM ok. So, here I am using the last 8 kilobytes of this one for EPROM. This is 8 kilobytes, this is for EPROM. So, the last address is F F F F H, what will be starting address if it is 8 kilo byte. So, in earlier also we have discussed about this how to I mean obtain knowing the starting address, how to know the ending address or knowing the ending address how to know the starting address. I will review the same thing again, 1 kilobyte is nothing but 1024 bytes ok.

1024 is decimal value what is hexadecimal equivalent of 1024, 1024 if you want to convert into hexadecimal you have to divide with 16. So, what will be this quotient and remainder? 1024 16 5s is 18, 16 6s is 96, then 96 remainder is 664, 4 is the quotient, 64 and remainder will be 0. Then, if I divide again 16, 16 4s are 64, 0 is the remainder, 16 4s are 0 is the 16 sorry, 0s 16 0s 4 is the remainder.

Now you have to stop here, until you get quotient as 0, then you have to read from bottom to top. So, 1024 decimal is equal to 4 0 0 hexa decimal. So, 1 kilobyte means total it will be 4 0 0 H. Suppose, if I start with 0 0 0 0 0 H 1 less than this, 1 less than this will be 0 0 3 F F H. This will represent 1 k. Because, 0 to 3 F F F means total 4 0 0 because you are including 0. So, 0 to 9 is 10, 10 numbers.

So, starting with 0, if you want 4 0 0 H, the ending address will be 1 less than this. 1 less than 4 0 0 is 3 F F because, if I add 1 to 3 F F, this will be F plus 1 is 1 0 F plus 1 is 1 0 4 0 0 ok. So, if I subtract 1 from 4 0 0 it will be 3 F F F.

(Refer Slide Time: 34:14)



So, we have to remember that now, 0 0 0 0 0 H to 0 0 3 F F H is 1 kilobyte ok. If I want 2 kilobytes, it will be 1 kilobyte plus 1 kilobyte ok. This is only 1 kilobyte. So, what is the next address of 0 0 3 F F H. So, this is the ending address starting of another kilobyte will be 1 plus this address. So, this will be 0 0 4 0 0 to get 1 k. So, for 0 0 0 0 H, we are adding 3 F F F to get 1 K. If the starting address is 0 0 4 0 0 H, 0 0 4 0 0 H for 1 K, the ending address will be you have to simply add 0 0 3 F F H.

So, this will be F F 7 0 0. So, the ending address will becomes 0 0 7 F F H only 1 F. So, this is another 1 K, if I start with 0 0 4 0 0 H. Then total from here to here is 1 K plus 1 K, 2 K which is 0 0 0 0 0 H to 0 0 7 F F H, this will be 2 K, this total this one will be 2 K. Similarly, if I want 4 K, this is 2 K 0 0 0 to 0 0 7 F F is 2 K, the next address to this one is 0 0 8 0 0 H, 0 0 8 0 0 H. So, how to get another 2 k? You have to add 7 F F F. So, 0 0 8 0 0 plus 0 0 7 F F. This will be F F F 0 0.

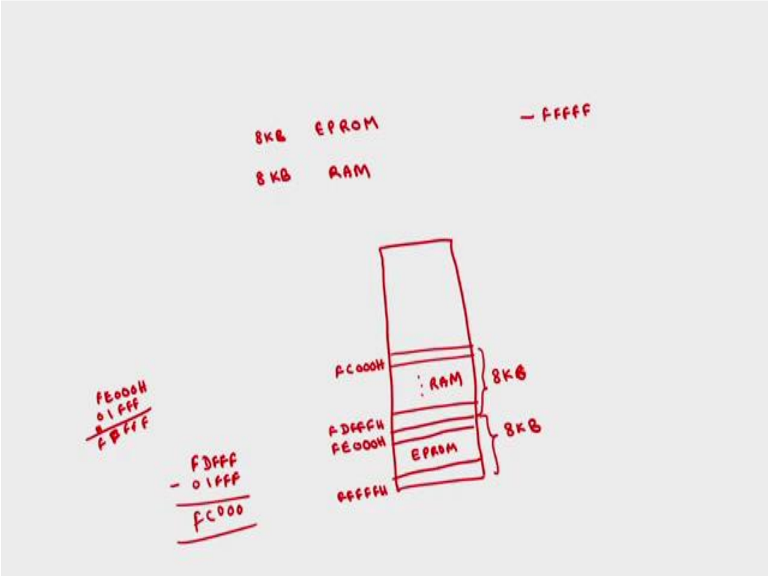
So, this to 0 0 F F F H will be another 2 K, if I start with 0 0 8 0 H this is another 2 K. So, if I start with 0 0 0 0 0 H and if I end with 0 0 F F F H, then this will be this 2 K, this 2 K, this is 4 K simply adding F F F will give 4 K. If I want 8 K so, what is the next address another 4 K, 0 1 0 0 0 H, if I add one to this 0 0 F F F H to get the next address F plus 1 is 10 F plus 1 is 10 F plus 1 is 1 0.

So, this will be 0 1 0 0 0 H and to get the ending address for the 4 k, you have to simply add F F F. If I add F F F here this will be 0 1 F F F H. So, ending address will be 0 1 F F F H, this is another 4 K. If I start with 0 0 0 0 0 H and if ending address is 1 F F F F H then this is total 8 K.

So, like that you can extend ok. So, this is for starting of 0 0 0 0 0 H, ending is 0 0 3 F F [FL], this is 1 K, here I am going to summarize 1 K, 2 K, 4 K, 8 K up to 8 K. So, normally, we will use maximum of 8 K or you can go for even higher also in a similar manner. So, 0 0 0 0 0 H to 0 0 3 F F F H will be 1 kilobyte and I will start with all zeros only 0 0 0 0 H to for 2 K it will be 0 0 7 F F H. This will be 2 K and if I start with same 0 0 0 0 0 H and if I end with for 4 K this will be 0 0 F F F H this will be 4 K I am starting with 0 0 0 0 0 H ending with 0 1 F F F H will be 8 K.

Similarly, if I want 16 K, this will be next address will be 0 2 0 0 0 plus 1 F F F, 0 1 F F F H this will be 0 3 F F F H. This will be 0 0 0 0 0 H to 0 3 F F F H will be 16 K and 32 K will be the next one is 0 4 0 0 0 to you have to add 0 3 F F F this will be 0 7 F F F. So, 0 0 0 0 0 H to 0 7 F F F H and 64 kilobytes we know 4 Fs. Starting with 0 0 0 0 0 H to 0 F F F F H. This is the table which will give 1 K, 2 K, 4 K, 8 K, 16 K, 32 K, 64 kilobytes starting with 0 0 0 0 0 H ok.

(Refer Slide Time: 41:09)



So, with this data. So, I come back to that interfacing of 8 kilobytes of EPROM, 8 kilobytes of RAM, random access memory. So, if I want to choose this as a last address last I mean a space of this one F F F F F H, the last portion of the memory. This 1 megabyte of memory, the last 8 kilobytes I want to use for EPROM.

This is 8 kilobytes. So, this ending address is F F F F F H. So, what in the starting address for 8 K it should be 1 F F F, we have to add 1 F F F. So, we have to subtract 1 F F F from here. So, 1 F F F means; this will be F E 0 0 0 H is it clear. So, F E 0 0 0 H, is the starting address to get the 8 K, we have to add 0 1 F F F H. So, this will be F F F 1 0 sorry, E F this will be F, so 4 Fs right.

Similarly, so, the address one below this one is this I want to use for another 8 K for RAM, Random Access Memory ok. So, what will be starting and ending address ending address will be F D F F F minus 0 1 F F F will be the starting address, this is ending address, ending address is only less than this which is F D F F F and starting address, you have to subtract 1 F F F will be 0 0 0 C F so, F C 0 0 0 H. So, these are the addresses of this RAM and EPROM ok.

So, to interface this RAM and EPROM in this locations, what did the hardware circuitry required and how this connections has to be made. So, I will discuss in the next class.

Thank you.