What is control unit.

- control operations of all parts of the computer
- but does not carry out any data processing operation.
- fetches internal instructions of the programs from the main memory to the processor IR, and based on this register contents, the CU generates a control signal that supervises the execution of these instructions.

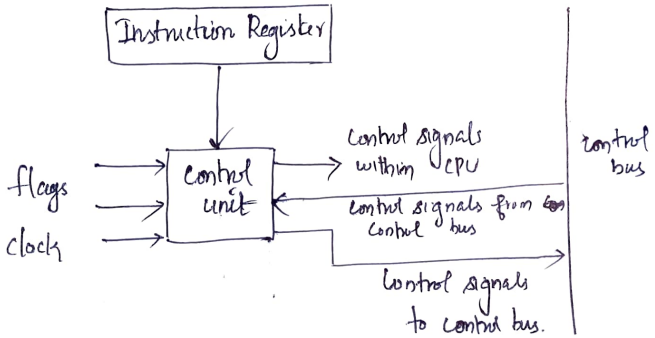


fig: Block diagram of the CU

Types of CU

Two approaches for generating control signals

(1) Hardwired ✓

(2) Microprogrammable control unit ✓

- ~~Cont~~ (1) Hardwired Control

- control signals are generated by specially designed h/w logical circuits - in which we cannot modify the signal generation method without physical change of the circuit structure.

(1) the OPCODE of an inst^n contains the basic data for control signal generation.

(2) In the Instruction Decoder, the opcode is decoded. The Inst^n decoder consist of a set of many decoders that decode different fields of the inst^n opcode

(3) The output lines going out of inst^n decoder obtains active signal values.

(4) output lines of inst^n decoder are connected to the inputs of the matrix that generates control signals for execution units of the computer.

(5) This matrix implements logical combinations of the decoded signals from the instⁿ opcode with the output from the matrix that generates signals representing consecutive control unit states and with signals coming from the outside of the processor.

(5) The Next control state Generator combines the o/p from the matrix and with the timing signals, which are generated by the timing unit

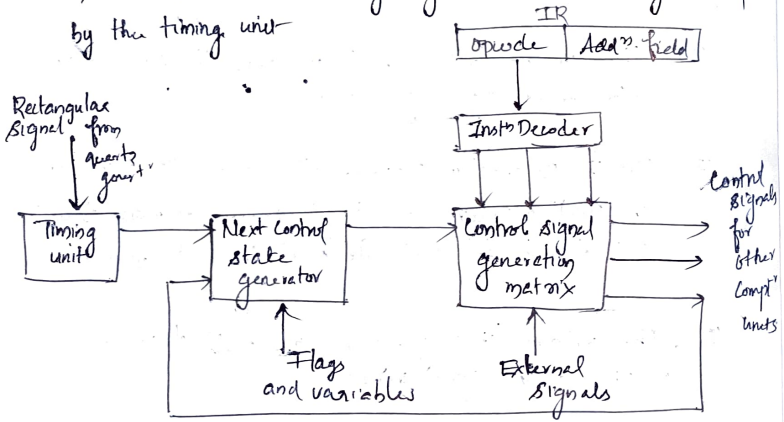


fig: Block diagram of a hardwired control unit of a computer.

Designing Methods of hardwired control unit

(1) state table method
(2) Delay element method
(3) Sequence counter method

In Hardwired CU - control signals are generated using hardware

(1) State Table method

T-states	I_1	Instructions I_2 ...		I_N
T_1	$C_{1,1}$	$C_{1,2}$	...	$C_{1,N}$
T_2	$C_{2,1}$	$C_{2,2}$	...	$C_{2,N}$
T_3	--	--	--	--
.	--	--	--	--
T_M	$C_{M,1}$	$C_{M,2}$	--	$C_{M,N}$

$C_{1,1}$ means control signal to be produced in T state (T_1) of instruction (I_1)

- Behaviour of the CU is represented ~~by~~ in the form of a table, which is known as state table

- each row represents T-state and the col. represents instructions

- col-to row intersection indicates which control signal will be produced in the corresponding T-state of an inst^n.

- Here, the h/w circuitry is designed for each column. for producing ctrl signals in different T-state

Adv

1) simplest method

2) mainly used for small inst^n set processors (RISC processors)

Dis

1) very large # of inst^{ns} in modern comptrs,
∴ the circuit bcmz complex, difficult to debug
If we make any modifications to the state table then the large parts of the circuit need to be changed. (not widely used)

(2) many redundancies in circuit design
Control signals for fetching the inst^n is common and which is repeated for N number of inst^n. So the cost of circuitry design may increase

(2) Delay Element Method

- control unit behaviour is represented in the form of flowchart.

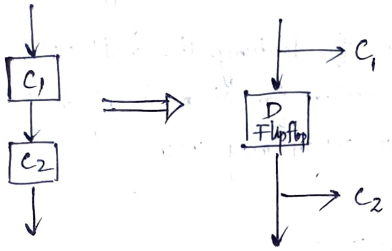
- Each step in the flowchart represents a control signal that needs to be produced for processing the inst^n

- If all the steps are performed, this means the inst^n is executed completely.

- Control signals perform micro-operations and each micro operation requires one T-state.

- For the micro operations which are independent, they are required to be performed in different T-state.
∴ for every consecutive control signals exactly 1-state delay is required, which can be produced with the help of D-FF.

- DFFs are inserted b/w every two consecutive ctrl signals.

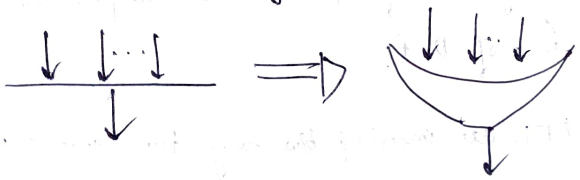


DFFs is introduced b/w each pair of control signals.

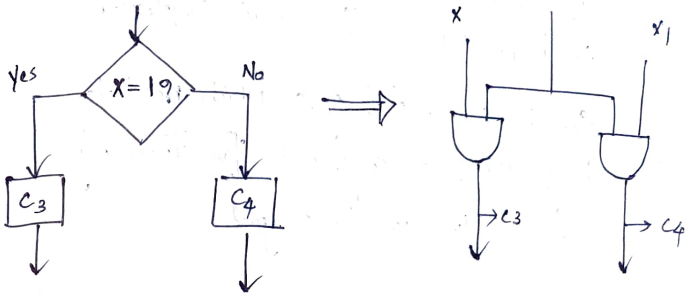
∴ After a ctrl sgl is generated, the delay element before the ctrl sgl is not in use until before the next inst" required that ctrl sgl.

∴ Among all D-flip flop's, only one will be active at a time. Therefore, this method is also known as one hot method.

* In a flowchart, if there ~~are~~ is multiple entry point for control signal then to combine two or more paths, we use an OR gate



Decision box is converted into a set of two complementing AND gates.



suppose the processor has two instructions add or subtract (∴ an opcode of 1 bit is needed in which '0' opcode for add inst" and 1 for subtract inst")

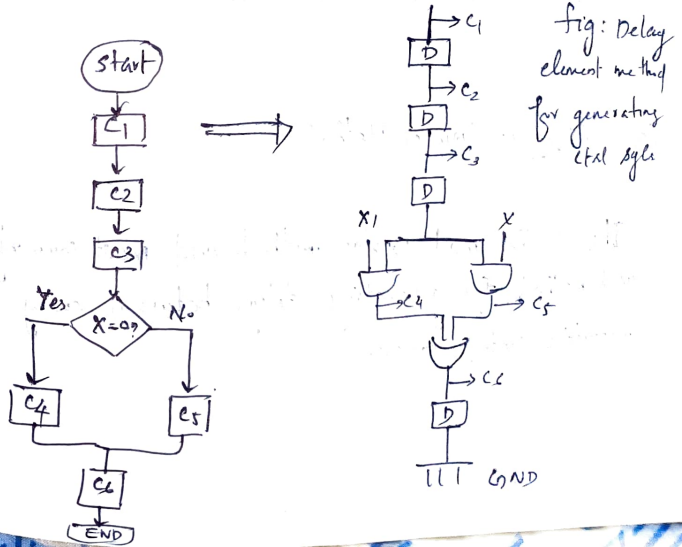


fig: Delay element method for generating ctrl sgls

Advantage

This method has a logical approach,
∴ helps to reduce the circuit complexity

for the common ctrl sgls, which need to be generated in every inst", for them only one circuitry can be designed.

Drawback

As the # of inst" increases, the # of DFF for generating delay is increased, so overall circuit complexity and cost increases.

(3) Sequence Counter Method

This is the most popular method and most commonly used method for generating delays between every consecutive control signals.

It's main advantage is,

uses logical approach of flowchart and doesn't use the unnecessary number of D FFs.

First, a flowchart is designed to represent the behavior of control unit.

It is then converted into a circuit using AND & OR gates.

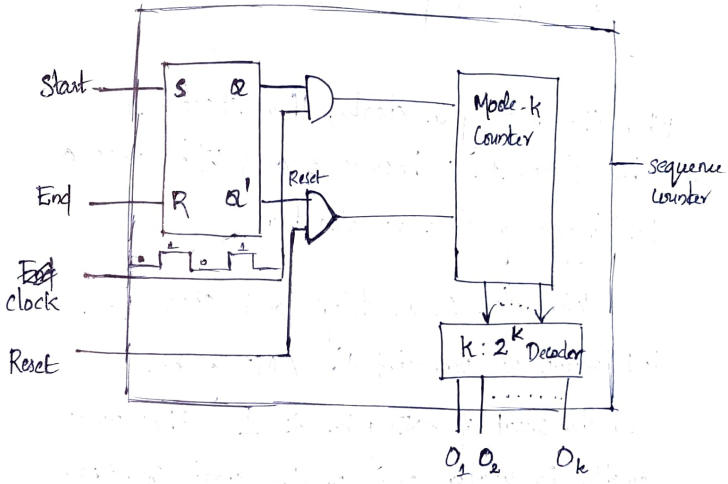


fig: Sequence Counter Method.

Similar to delay element method.
the difference - instead of unnecessary D FFs, there are triggering points in the circuit.

Working of Sequence counter Circuit

Clock	S	R	Q	Q'
0	X	X	Memory	
1	0	0	Memory	
1	0	1	0	1
1	1	0	1	0
1	1	1	Not used	

- When the inst^n cycle starts, then start = 1
- When start = 1, Q becomes 1 and Q' becomes 0
- Level triggering clock is also here

When clk = 1 or high and start = 1, both outputs are connected to AND gate, so the output of AND gate is 1, that will enable the counter and counter starts counting from 000 state. so the 000 state is decoded by the decoder and produces output O_1, which will trigger the triggering point in the CU.

- As the clk becomes high again after 1 T-state ∴ When clk = 0, the counter state is preserved remains the same until the clk being high again. This make sure that the counter changes its states after a gap of one-one T state.

- If the counter is of k bits then k : 2^k decoder is required this can produce 2k outputs and that will trigger the 2k triggering points after a gap of 1-1 T-states in the circuit.

- ~~If reset~~ End pin = 1, then the counter will reset and ~~after that it will again~~ start count

- When the inst^n ends, the control signal is generated to make End pin = 1, and the counter is reset, so the next time, it begins from the first count (000)

Advantages

Less # of FFs are used.

Microprogrammed Control Unit

- It is a CU where binary control variables are stored in memory.
- The control function that specifies a microoperation is a binary variable.
- Binary variables at any time can be represented by strings of 1's and 0's. called control word
- Each control word contains a micro instruction
- Microinstruction specifies one or more microoperations.
- Sequence of microinstructions constitute a microprogram

A computer with microprogrammed control unit will have two memories. Main m/y and control m/y

Main m/y - available to user for storing programs and its contents chang when data are manipulated

Control m/y - holds a fixed microprogram that can't be altered by user and it contains instns that specifies control signals.

- Control memory can be ROM whose contents are fixed.

Some important Terms.

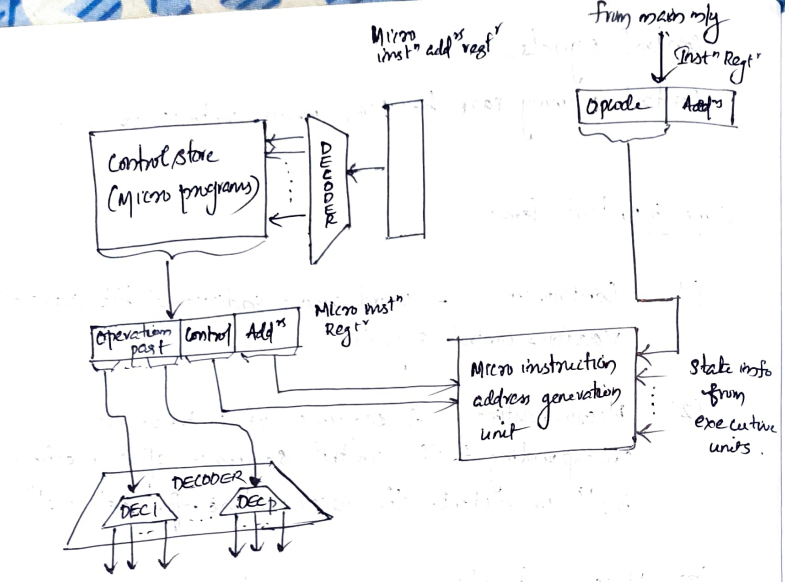
(1). Control word - word whose individual bits represents various control signals.

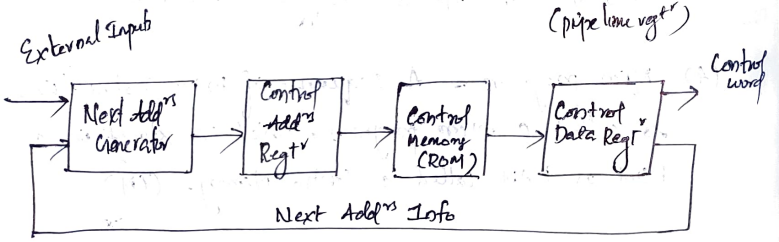
(2) Micro routine - A sequence of controlwords corresponding to the control sequence of a machine inst^n constitute the microroutine for that inst^n.

(3) Micro instruction - Individual control words in this micro routine are reffered to as micro inst^n

(4) Micro program - A sequence of micro instructions is called a micro program, which is stored in a ROM or RAM called a Control Memory. (CM)

(5) Control Store - The micro-routines for all instructions in the instruction set of a computer are stored in a special memory called the control store.





Microprogrammed control.

Desired setting of the control signals in each step determined by a program stored in a special mly.

- The control pgm is called a microprogram.
- stored on the processor chip in a small and fast mly called the control store.
- Controlword - words whose individual bits represents control signals for each step in sequence.

Microroutine — sequence of control words corresponding to control sequence for a machine inst".

- individual control words are referred to as micro instructions.
- Microroutines for all machine inst" are stored in the control store.

Micro instructions	PC in	PC out	MAR in	Read	MDR out	IR in	Y in	Select	Add	Z in	Z out	R1 in	R3 out	MFC	End
1	0	1	1	1	0	0	0	1	1	1	0	0	0	0	0
2	1	0	0	0	0	0	1	0	0	0	1	0	0	1	0
3															
4															
5															
6															
7															

Microprogram Sequencer

Microprogram control unit — two parts

(1) control mdy - stores the μ instns

(2) associated ckt - controls the generation of next addrs

Address generation part is sometimes called Microprogram Sequencer. Since it sequences the microinstructions in control memory.

A μ program sequencer attached to a control memory inspects certain bits of the microinstn, from which it determines the next addrs for control memory. A typical sequencer provides the following addrs-sequencing capabilities.

(1) Increments the present addr

(2) Branches to an addr as specified by the addr field of the μ instruction.

3. Branches to a given add^{n} if a specified status bit is equal to ▲ 1.

4. Transfers control to a new add^{n} as specified by an external source.

5. Has a facility for subroutine calls and returns.

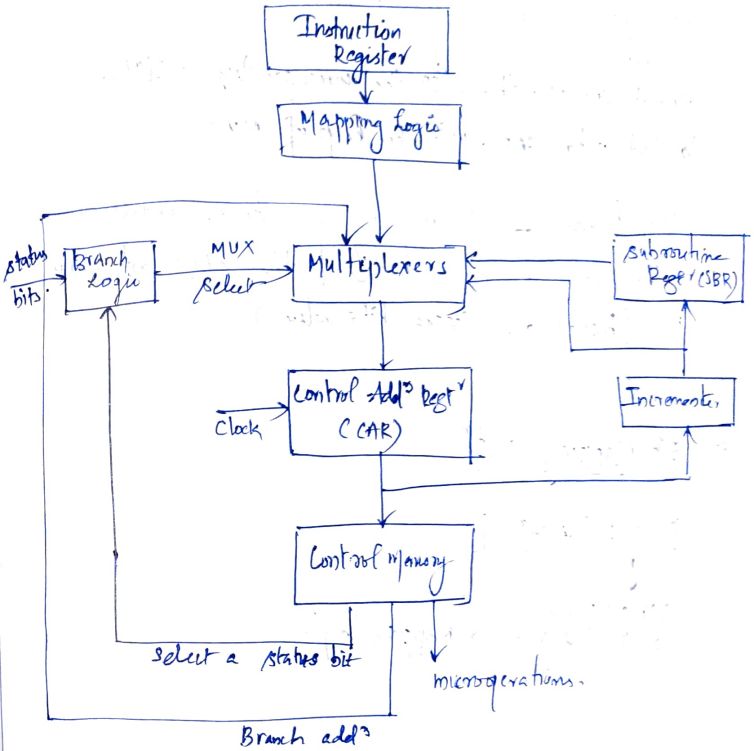
How we can generate the next ~~add^{n}~~ instruction. add^{n}

The duty of μ program sequencer is to generate the next inst^{n} add^{n}.

Four approaches.

1) Incrementing CAR by 1.
   [CAR] → add^{n} of μ inst^{n}.

2) Subroutine Call and Return (SBR)



- Subroutine - just a function
- Once the exct^{n} of subroutine is over the ctrl comes to the next statement of the main Pgm.

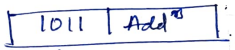
3) unconditional or conditional Branch

↓

without checking the cdtn the corresponding addrs is loaded into CAR. with the help of MUX

Conditional — whenever the cdtn is satisfied then only we have to load the addr into CAR. (based on status bits value)

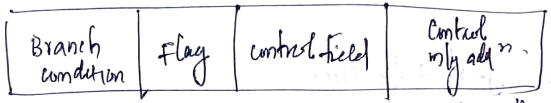
4) Mapping inst^n



↓

0X xxx 00

0101100 ⟶ mapping logic



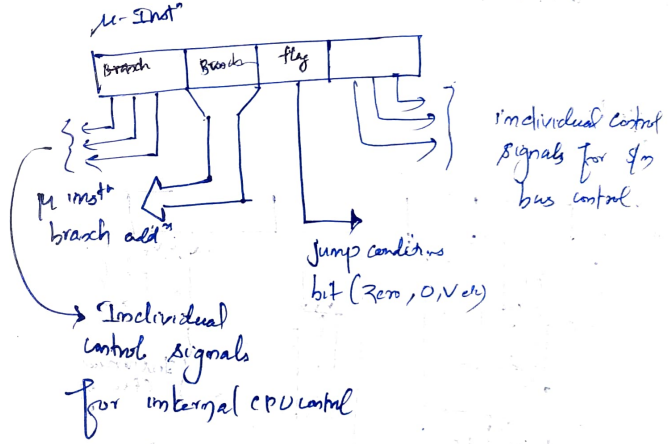
$\log_2 16$ for horizontal 64 bits/cs $\log_2 2^{12} = 12$ bits.

for vertical it $\lceil \log_2 64 \rceil = 6$ bits/cs

Micro Instruction format

(1) Horizontal
(2) Vertical
(3) Hybrid

1) Horizontal



→ there will be no external circuits (decoder, demux)

→ each bits in the inst^n controlbits its own purpose

→ faster execution

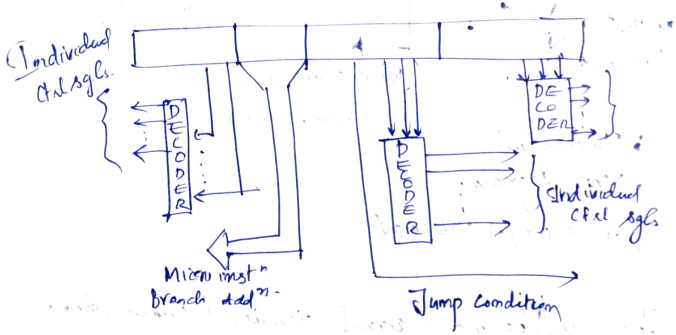
ADD
SUB
MUL
DIV
} 4 instns.

R_0 to R_{n-1} - 8 registers

MUL R_3, R_4 . $R_3 \leftarrow R_3 * R_4$

0010 $\overset{R_1}{00100000}$ $\overset{R_2}{00010000}$

Vertical



Decoder: Combinational circuit

n input ✓

2^n o/p ✓

MUL R_2, R_3.

10 010 011 20 bits

ADD 00 R_0 000
SUB 01
MUL 10
DIV 11 R_7 111

# of bits are less
costly circuit
delay in decoding
by consumption low.

Suppose there are 64 control signals in the system and 1024 words control memory. Find out the size of horizontal and vertical μ instns 2^6

H

# of cs = 64

# of bits required to represent the 64 control signals.

$= \lceil \log_2 64 \rceil = 6$

# of cw in the CM = $1024 = 2^{10}$

# of addn bits = 10

Size of μ instn = bits for cs + # of addn bits

$= 64 + 10 = 74$ bits

Size of vertical μ instn = bits for cs + # of addn bits

$= 6 + 10$

$= 16$ bits

Consider a µ programmed CU, where 1024 words control mlg is used. The CU has to support 50 control signals and 16 flag conditions. How many bits are required in the CW? what is the size of the control memory.

# of control signals = 50

# of bits required to represent the 50 control signal = $\lceil \log_2 50 \rceil = 6$ $\quad 2^5 < 50$

# of control words in the CM = $1024 = 2^{10}$

# of addr bits = 10

# of flag cdtns = 16

# of bits to represent flag cdtn = $\lceil \log_2 16 \rceil$

= 4 //

Size of horizontal µ instn = bits for control sgls + bits for flag cdtns + # of addrs bits

= 50 + 4 + 10 = 64 bits.

Size of vertical microinstruction.

= bits for control signals + bits for flag cdtns + # of addrs bits

= 6 + 4 + 10 = 20 bits

Size of horizontal control mlg

= # of control word in memory * size of horizontal µ instn

= 1024 * 64 bits

= 1k x 64 bits

= 1k * (8*8) bits

= 1k x 8 bytes.

= 8KB.

Size of vertical control mlg

= # of control word in mlg * size of vertical microop instn

= 1024 * 20 bits

= 1k * 3 bytes

= 1k * 3 bytes

= 3KB

There are 2 types of microprogrammed CU on the basis of control words., which are saved in the control mfy.
These microprogramed control units are the vertical microprogrammed CUs and Horizontal μprogrammed CUs.

Horizontal microprogramming

- the cs are represented in decoded binary format that is one bit/ one control signal.

If 53 cs are present in the processor then 53 bits are required. This it's supports longer control words.

- It requires no additional decoder beq one control signal is needed to be decoded. It is faster than vertical μ programmed ctrl unit.

- more than one control signal can be enabled at a time. this wit allows higher degree of parallelism

- Horizontal μ progrg offers great flexibility beq each control bit is indepen dent of each other.
It has greater length so it typically contains more info than vertical microinst".

- each μ inst" is capable enough to control several resources simultaneously, it has the potential advantage of more efficient h/w utilization.

2) Vertical μ programming.

employs a variable format and high degree of encoding, as opposed to horizontal microprogramming. It supports shorter control word than horizontal control unit. A code is used for each microoperation to be performed. For N signals $\lceil \log_2 N \rceil$ bits are required.

(decoder)

→ It requires an additional h/w to generate controlsgls
- It is slower than horizontal control unit

$2^1 \times 2^8$
larger

Consider a hypothetical control unit that supports 4k words. The Hardware contains 64 control signals and 16 Flags. What is the size of control word used in bits and control mly in a byte using

a) Horizontal prgng
a) Vertical prgng

Branch cdtn	Flag	Control fields	control mly access

$\log_2(4*1024)$

$\log_2(2^2 \cdot 2^{10})$

$\log_2 2^{12}$

12 bits

a) For Horzlt

64 signal ⟶ 64 bits

16 bits ⟶ 16 Flags

Control word size (# of bits in cw) = 64 + 16 = 80 bits

Control mly = 4k w = (4 * 80) / 8 = 40 k Byte

b) for vertical

6 bits for 64 signals. → $\lceil \log_2 64 \rceil = 6$

flags $\lceil \log_2 16 \rceil = 4$.

12 bits for 4k words

$\log_2(4*1024)$

control word size = 4 + 6 + 12 = 22 bit

control mly : 4 kW = (4 * 22) / 8

= 11 k Byte.