
DO NOT SHARE
SLIDES AND CLASS MATERIALS
ON ONLINE SITES
Course Hero

Advanced Logic Design
Microprogrammed Control

Mingoo Seok
Columbia University

Slide source: Barry Watson, Stockholm;
D. Mukhopadhyay IIT Madras;

Example: GCD Processor

gcd(a,a) = a

gcd(a,b) = gcd(a-b, b) if a>b,

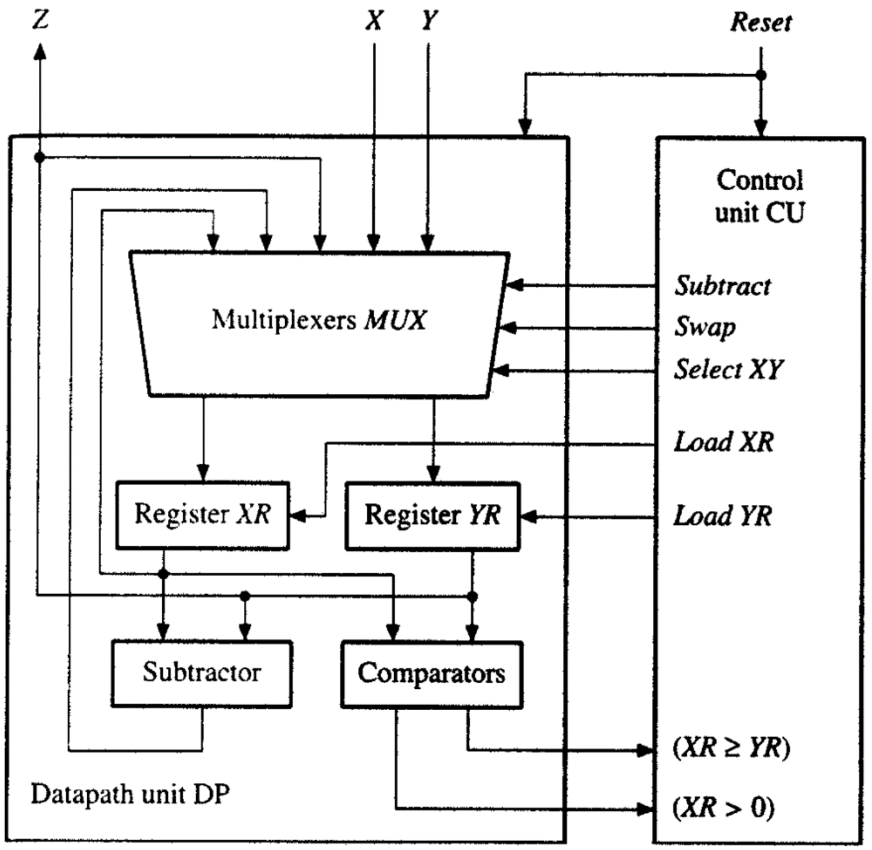
gcd(a,b) = gcd(a, b-a) otherwise.

Hardware for the GCD Processor

gcd(a,a) = a

gcd(a,b) = gcd(a-b, b) if a>b,

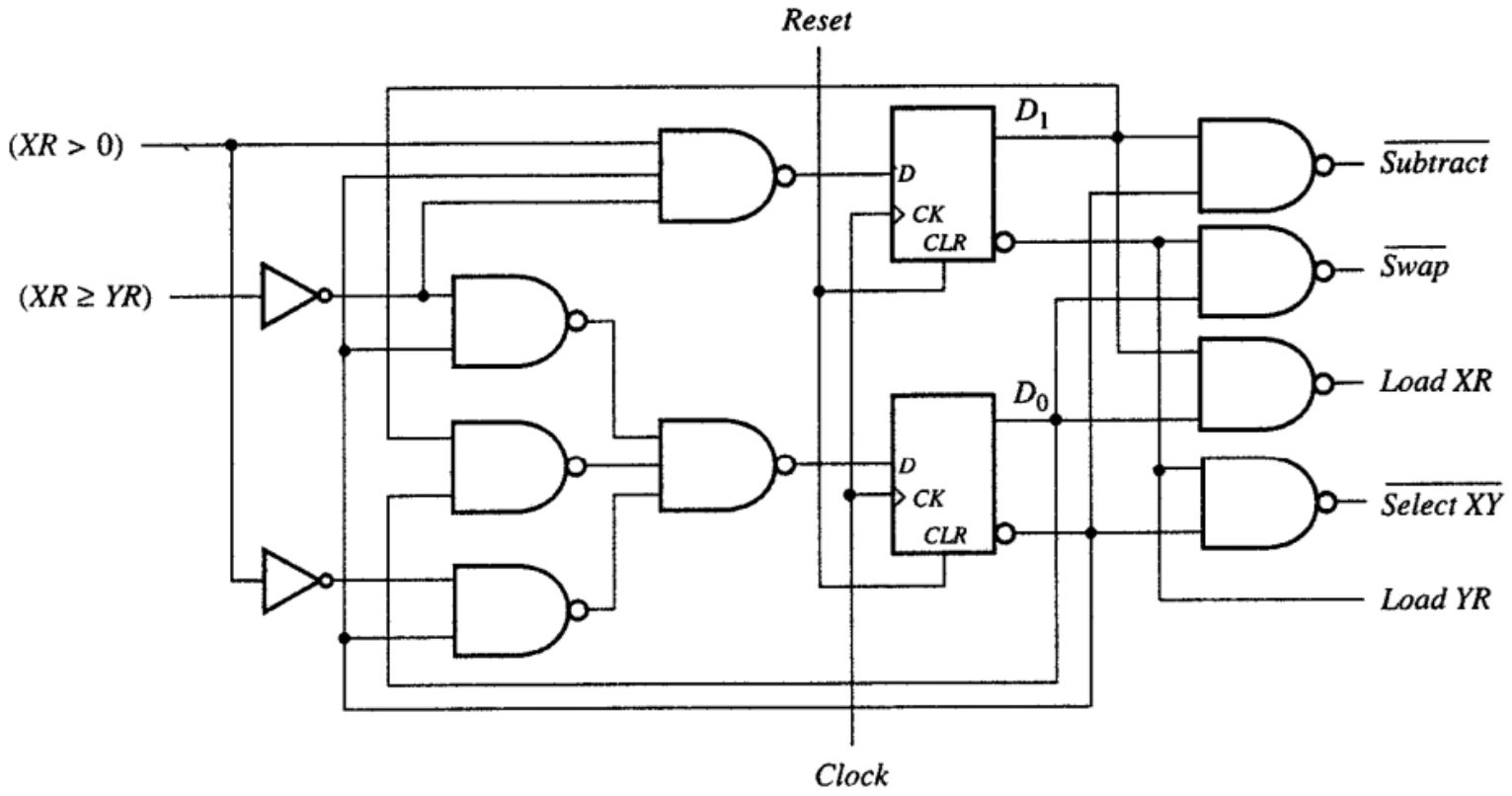
gcd(a,b) = gcd(a, b-a) otherwise.



State Table of the Control Unit

	Inputs ($XR > 0$) ($XR \geq YR$)			Outputs				
State	**0–**	**10**	**11**	***Subtract***	***Swap***	***Select XY***	***Load XR***	***Load YR***
S_0 (Begin)	S_3	S_1	S_2	0	0	1	1	1
S_1 (Swap)	S_2	S_2	S_2	0	1	0	1	1
S_2 (Subtract)	S_3	S_1	S_2	1	0	0	1	0
S_3 (End)	S_3	S_3	S_3	0	0	0	0	0

Hardwired Control



Microprogram: Concept

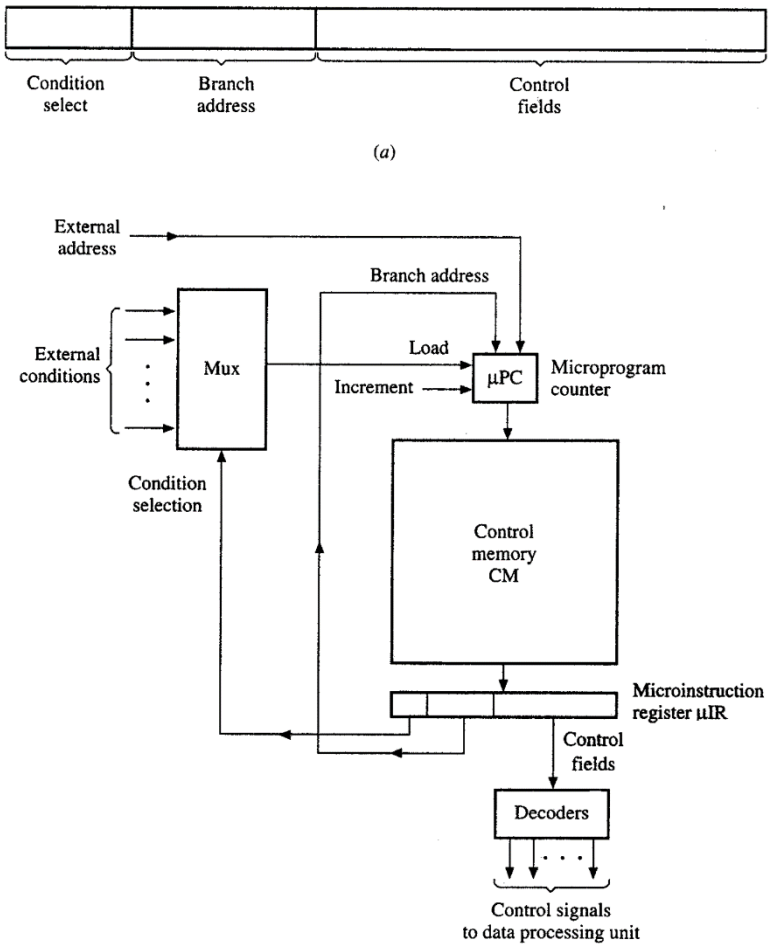
- High level description of a double precision ADD:
  - ADD AL, BL
  - ADDC AH, BH
- Low level description: microprogram

Cycle	Function Select	Storage Control	Data Routing
1	Add	Read AL, Read BL, Write AL	…
2	Add with carry	ReadAH, Read BH, Write AH	…

Microprogram: Definition

- Microprogram
  - Program stored in memory that generates all the control signals required to execute the instruction set correctly
  - Consists of microinstructions
- Microinstruction
  - Contains a control word and a sequencing word
  - **Control word**: all the control information required for one clock cycle
  - **Sequencing word**: information needed to decide the *next* microinstruction address
- Control memory (control storage)
  - Storage in the microprogrammed control unit to store the microprogram

Microprogramed Control



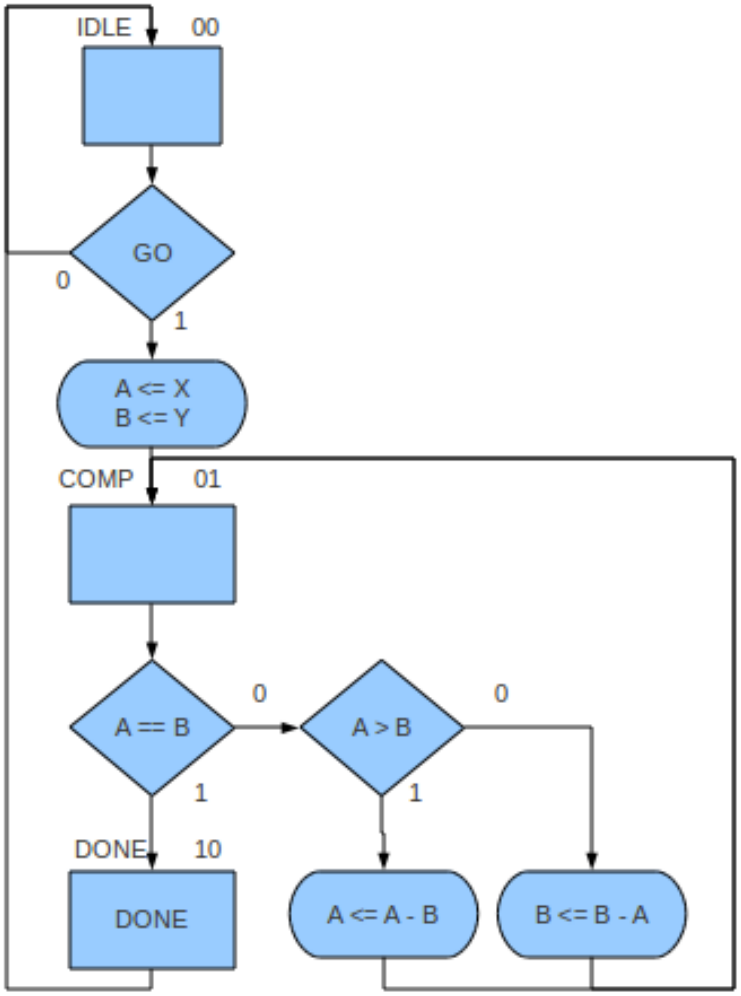
Example: GCD Processor

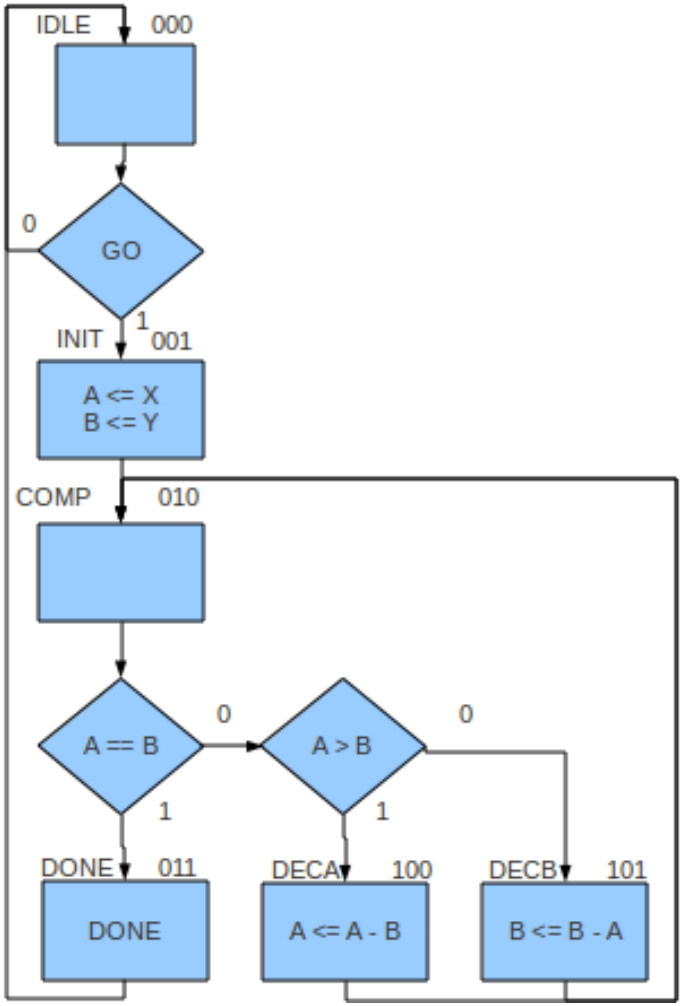
gcd(a,a) = a

gcd(a,b) = gcd(a-b, b) if a>b,

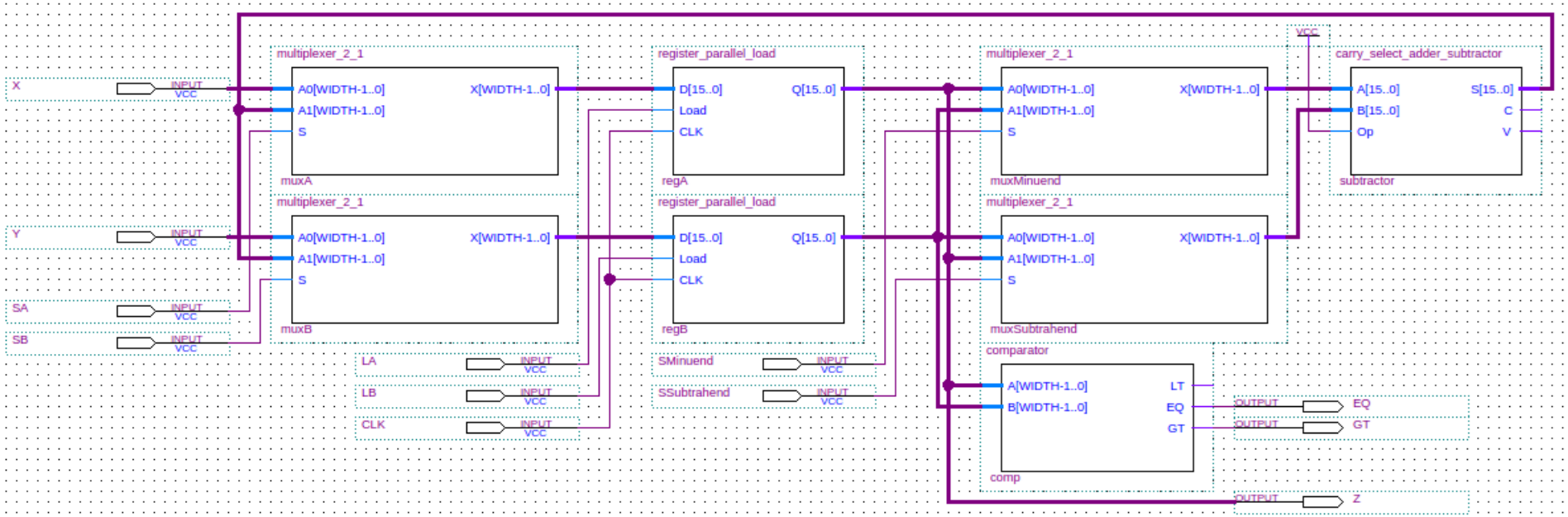
gcd(a,b) = gcd(a, b-a) otherwise.

ASM: Mealy vs Moore





Datapath


multiplexer_2_1
A0[WIDTH-1..0]
A1[WIDTH-1..0]
S
X[WIDTH-1..0]
muxA
register_parallel_load
D[15..0]
Load
CLK
Q[15..0]
regA
multiplexer_2_1
muxMinuend
carry_select_adder_subtractor
A[15..0]
B[15..0]
Op
S[15..0]
C
V
subtractor
VCC
X
Y
SA
SB
INPUT
VCC
multiplexer_2_1
muxB
register_parallel_load
regB
multiplexer_2_1
muxSubtrahend
LA
LB
CLK
SMinuend
SSubtrahend
comparator
A[WIDTH-1..0]
B[WIDTH-1..0]
LT
EQ
GT
comp
OUTPUT
EQ
GT
Z

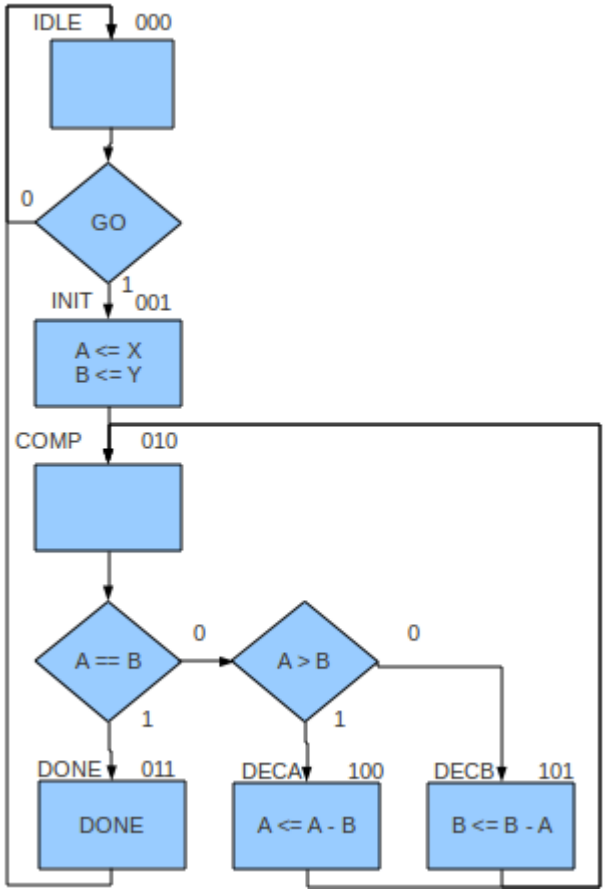
Microprogram

- The ASM now comprises six states (IDLE, INIT, COMP, DECA, DECB, and DONE). This increase in the number of states will impact computation time as each state transition takes a clock period.
- We have six states so we will need six addresses, therefore a ROM address will comprise three bits ($2^2=4 < 6 < 2^3=8$)

Branch

- We'll also encode the next microinstruction address in the current microinstruction being executed. This means that the instruction will need an address to jump to for each possible combination of the three condition inputs (GO, EQ and GT).
- In the worst case, each instruction can be followed by one of the eight instructions. This requires each instruction needs to contain 8 (2^3) addresses. Since each address is three bits, we may need 24 bits just for addresses. Too much!
- Looking at the ASM we see that we can get away with less than half of that. In the instruction that checks GO we only need ***two*** addresses: the address to jump to on GO == 0 and the address to jump to on GO == 1. In the instruction that will compare A and B we only need ***three*** addresses: EQ == 0 & GT == 0, EQ == 0 & GT == 1, and EQ == 1 & GT == 0.
- Since we can't execute these two instructions at the same time we can get away with the worst case which is three target addresses, so we need only ***3 addresses (9 bits)***.
- We'll call these three target addresses Next00, Next01, and Next10. We'll also use a signal SEL to tell us how to interpret these three addresses.

Branch



SEL	Next00	Next01	Next10
0	GO == 0	GO == 1	Unused
1	EQ == 0 & GT == 0	EQ == 0 & GT == 1	EQ == 1 & GT == 0

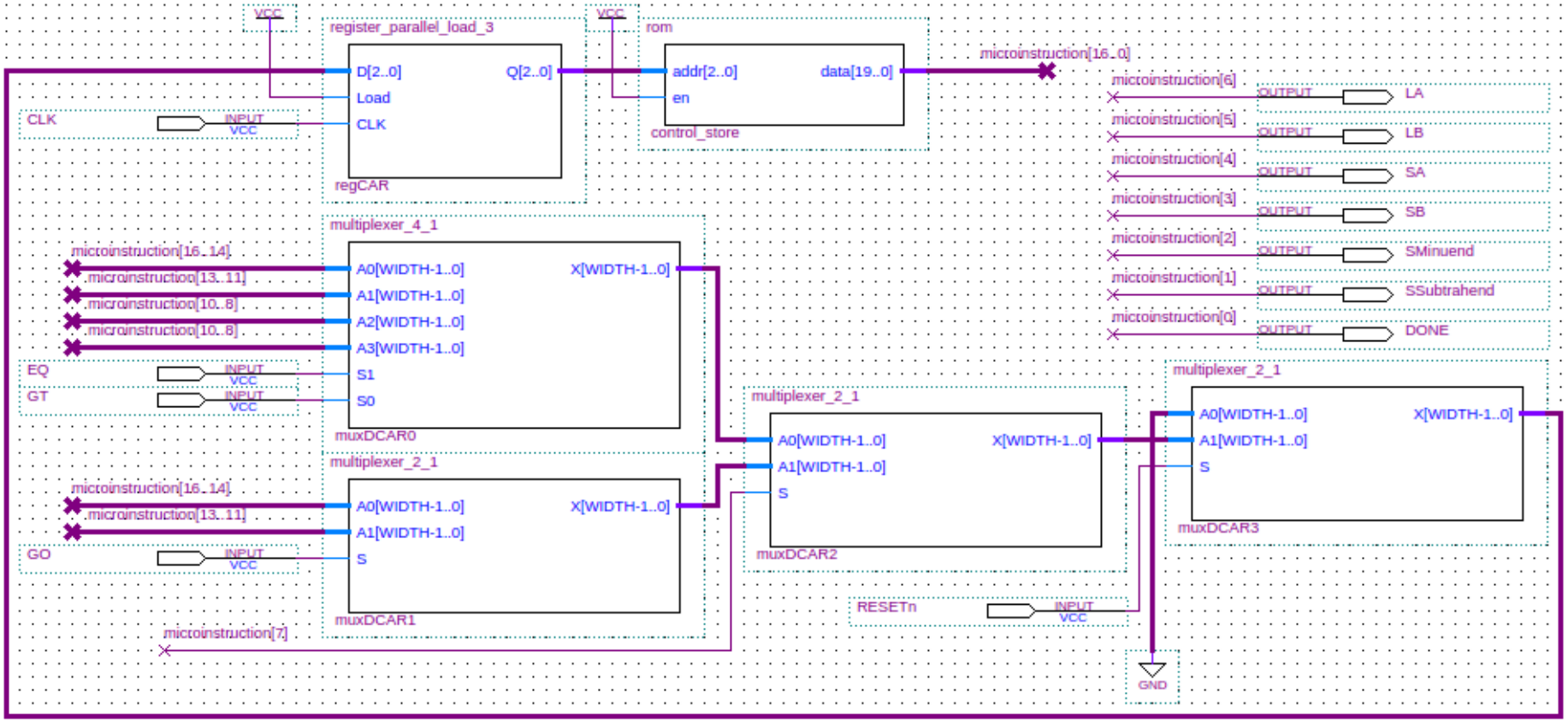
Datapath Control Signal & Control Unit Output

- For the datapath control signalling we need six bits: LA, LB, SA, SB, SMinuend, and SSubtrahend. These control signals are Moore outputs so they depend only upon control unit state (or current control store address).
- For the control unit output we need only one bit: DONE. Again, this is a Moore output that only depends upon control unit state (or current control store address).

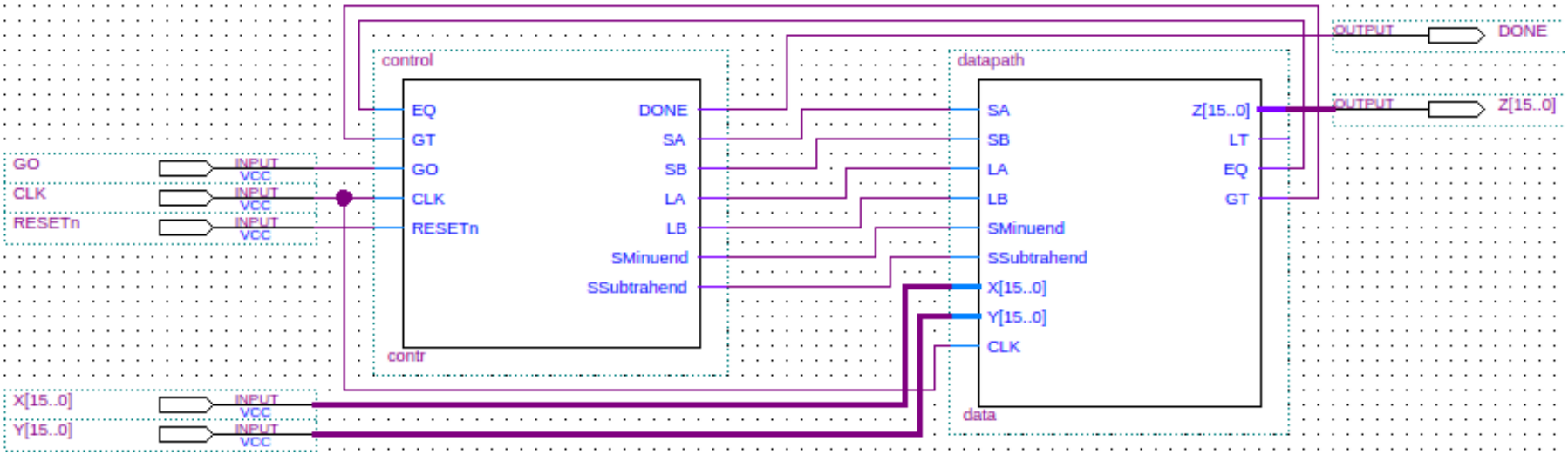
Control Storage

	Control Word										
	Target Addresses				Datapath Control						Control Unit Output
Address	Next 00	Next 01	Next 10	SEL	LA	LB	SA	SB	SMinuend	SSubtrahend	DONE
000	000	001	000	1	0	0	0	0	0	0	0
001	010	010	010	1	1	1	0	0	0	0	0
010	101	100	011	0	0	0	0	0	0	0	0
011	000	000	000	0	0	0	0	0	0	0	1
100	010	010	010	0	1	0	1	0	0	0	0
101	010	010	010	0	0	1	0	1	1	1	0

Microprogrammed Control



Datapath + MP-ed Control



Full Verilog code:
http://barrywatson.se/dd/dd_microprogrammed_nonprogrammable_control_unit_gcd.html

Comments

- Microprogramming helps in making a control unit which may be changed by changing the content of the memory
- As compared to a hard-wired control unit, it is slow due to the fetch timing of the instruction from the memory