ECE378 Final Project (Winter 2015)

PROJECT GUMMI

16-BIT MICROPROCESSOR

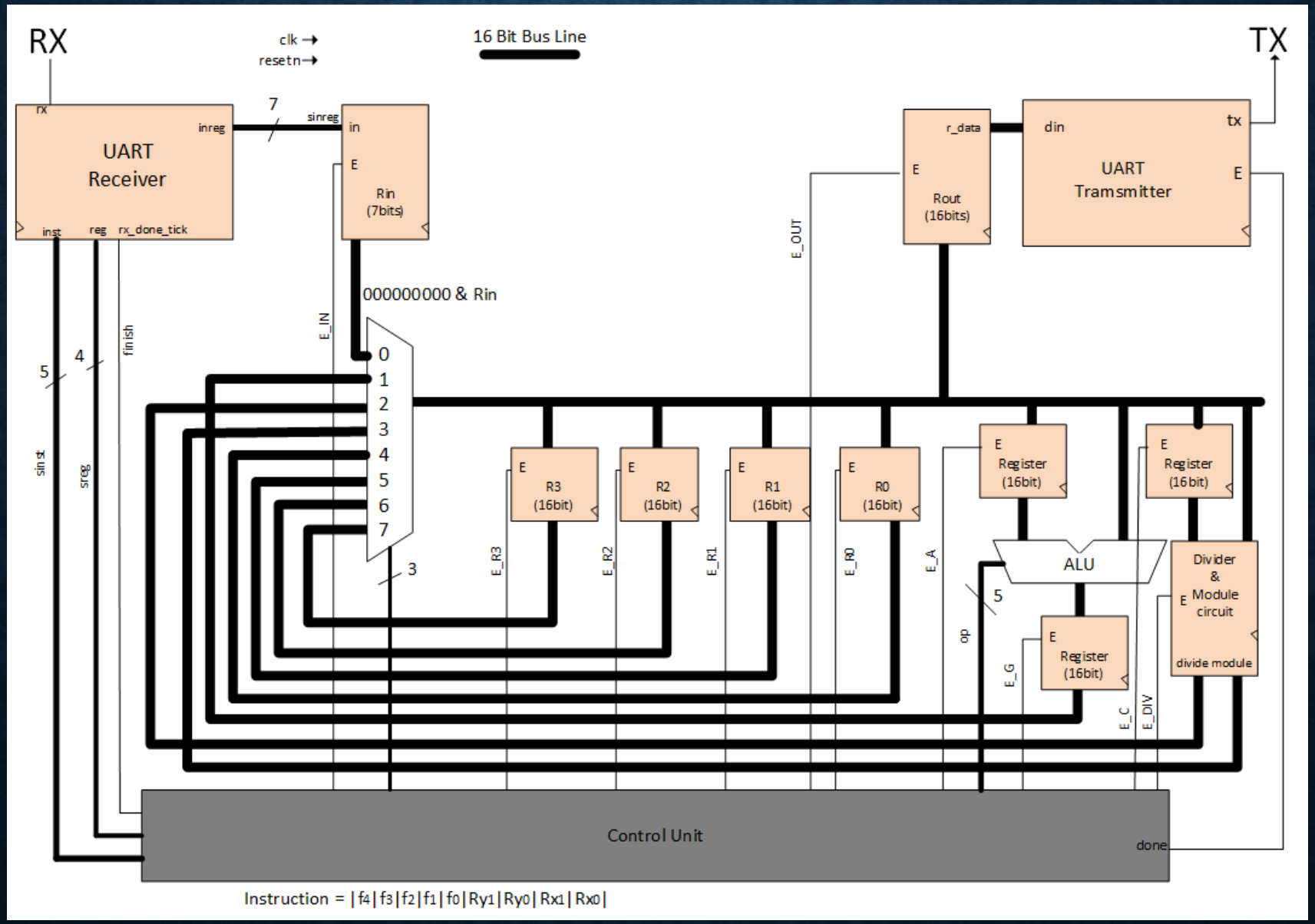

Kazumi Malhan


Chris Petros

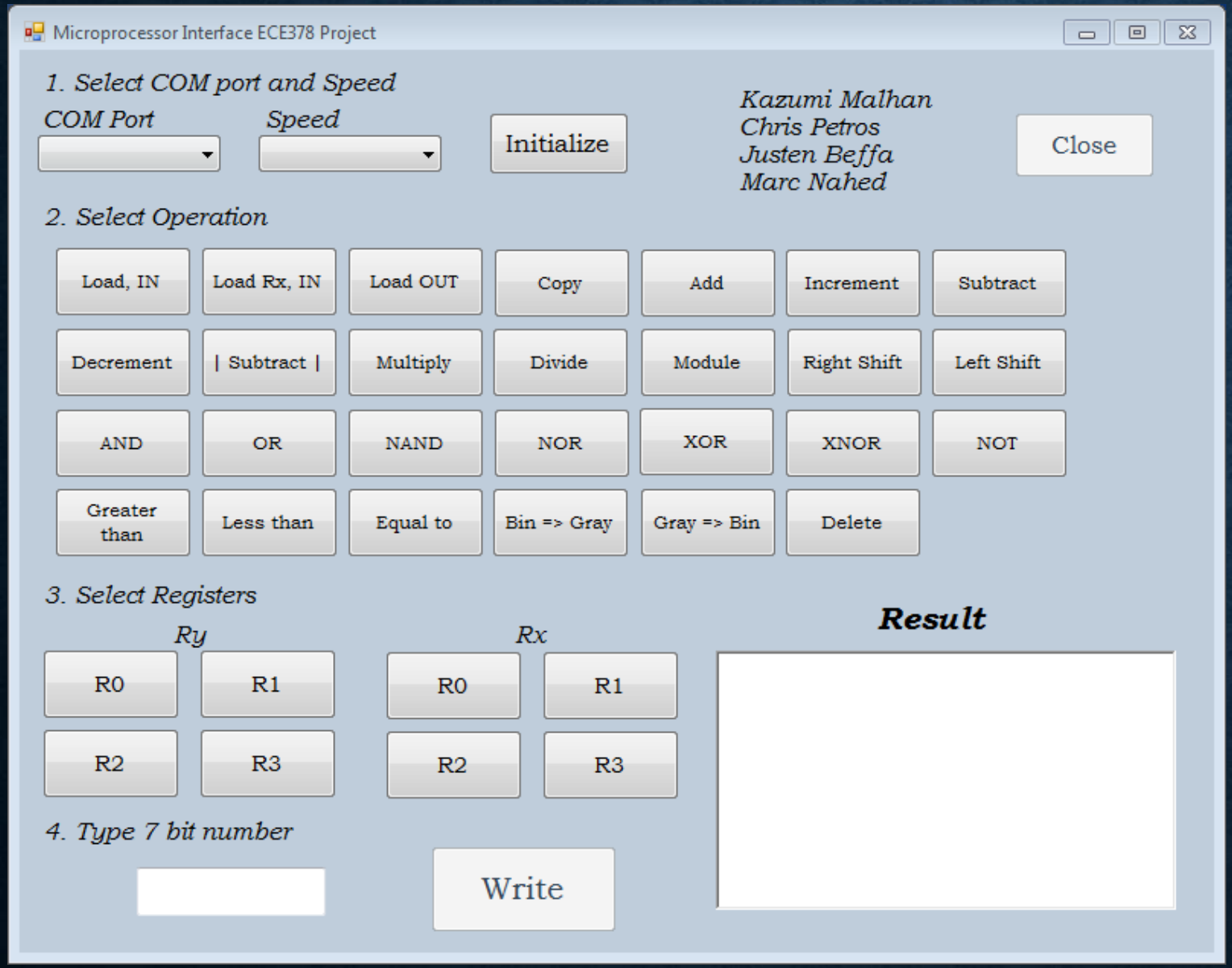

Justen Beffa


Marc Nahed

TOP LEVEL DESIGN



SERIAL INTERFACE ON PC SIDE



Developed using Visual Basic

Users can only select one button from each category

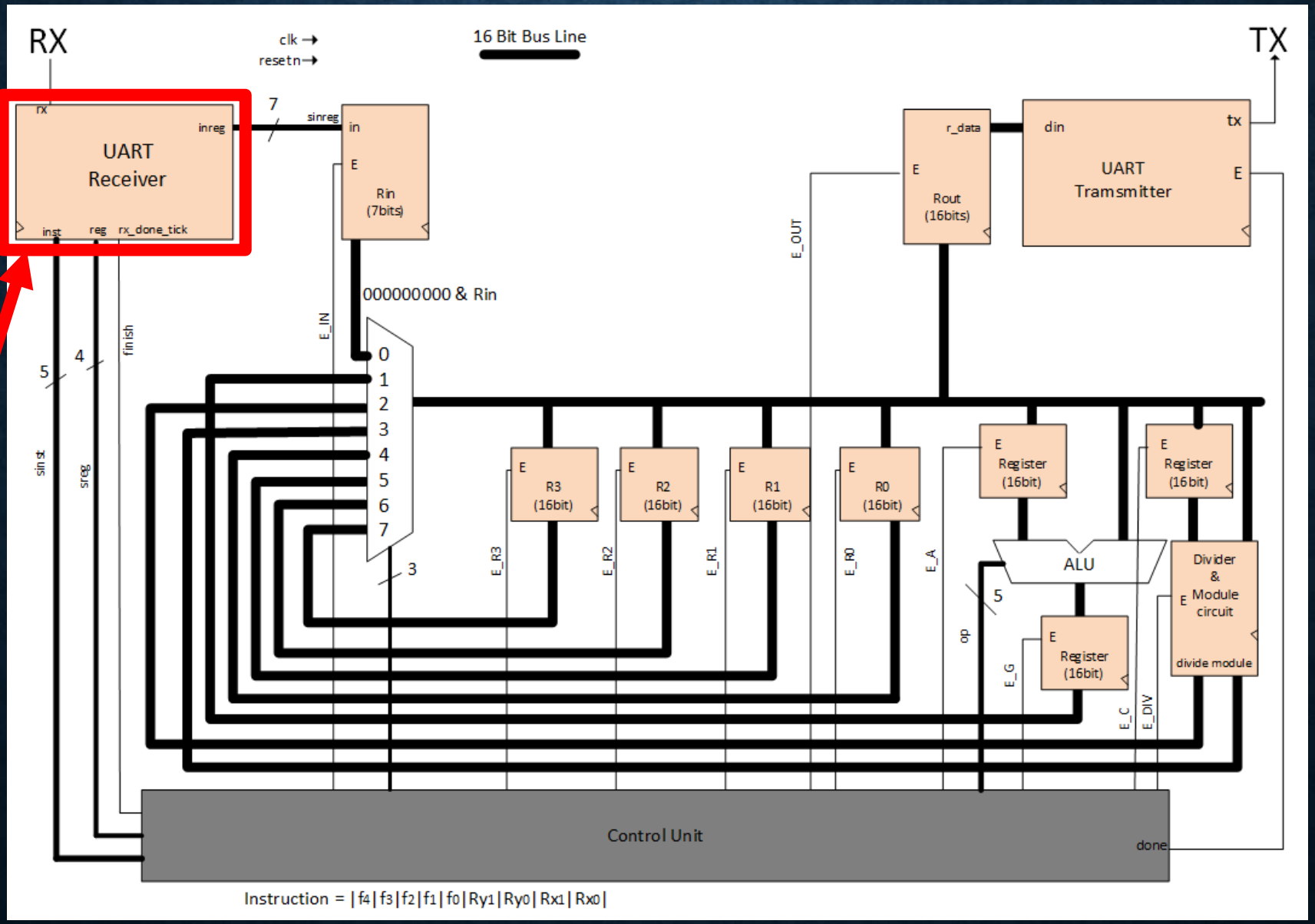
Write button is enabled only after connection is established

Sending appropriate ASCII code that matches to the instruction set

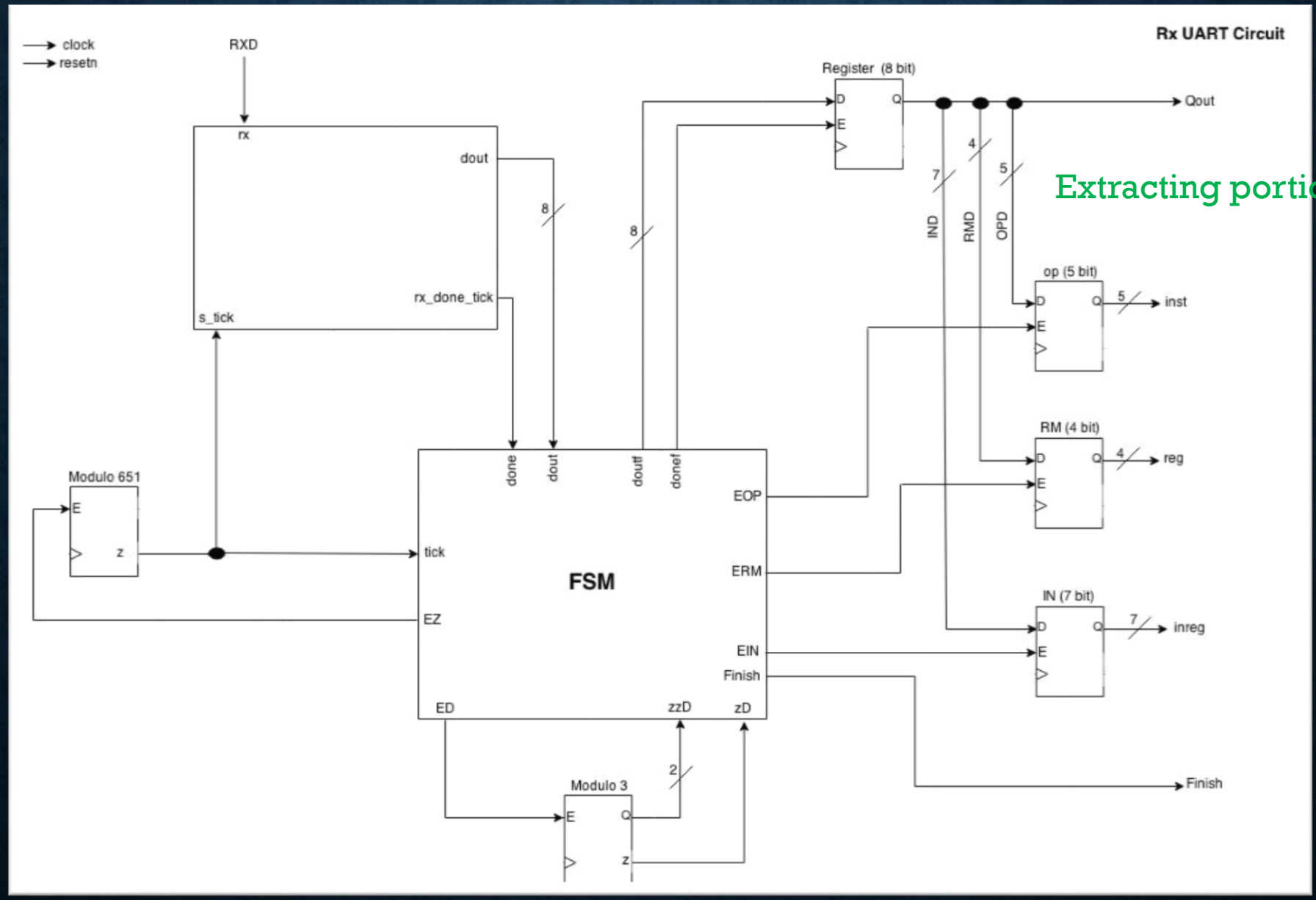
Users can only type 1 or 0 up to 7 bits to input box

Users cannot write on result box

TOP LEVEL DESIGN



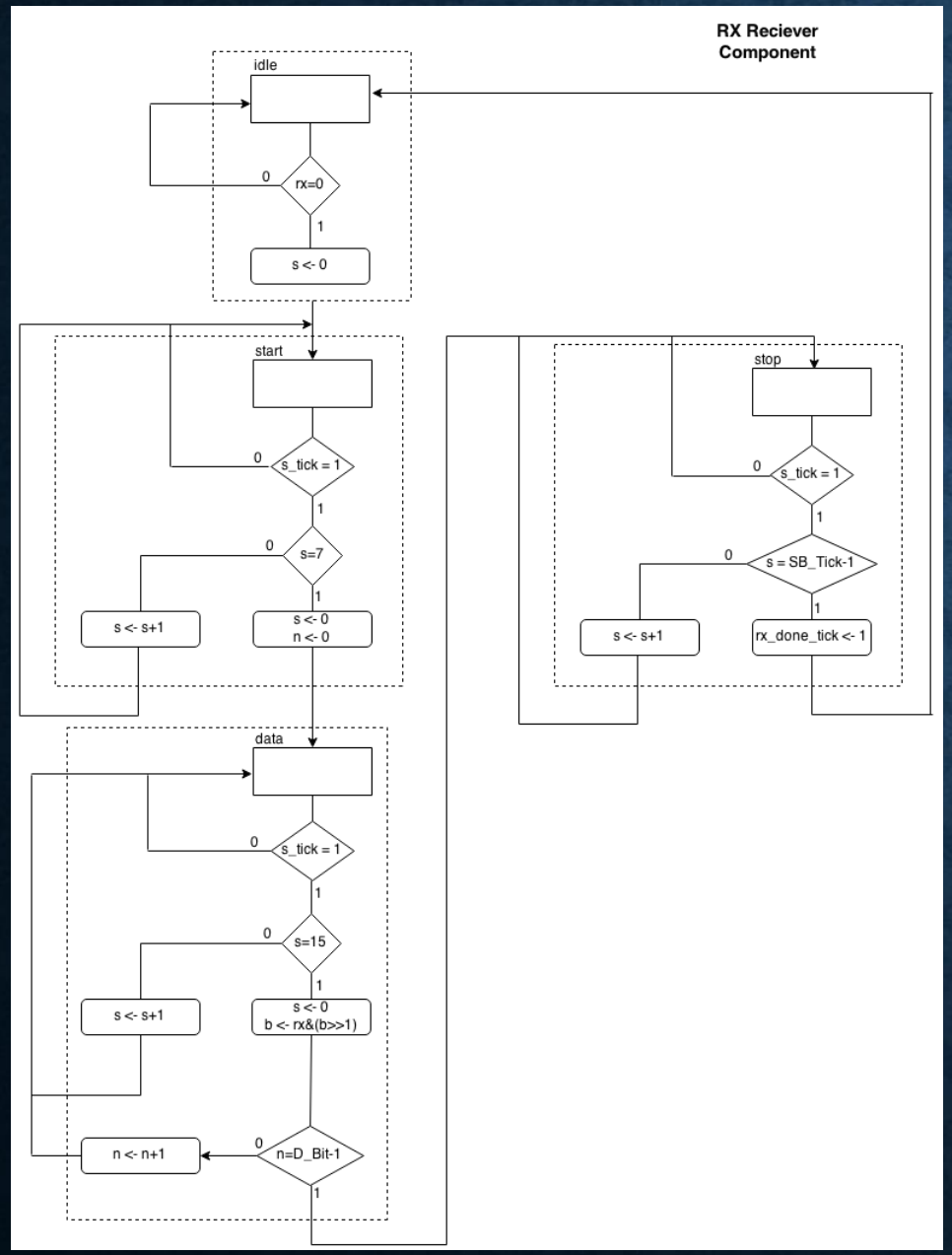
UART (RX) DATA PATH



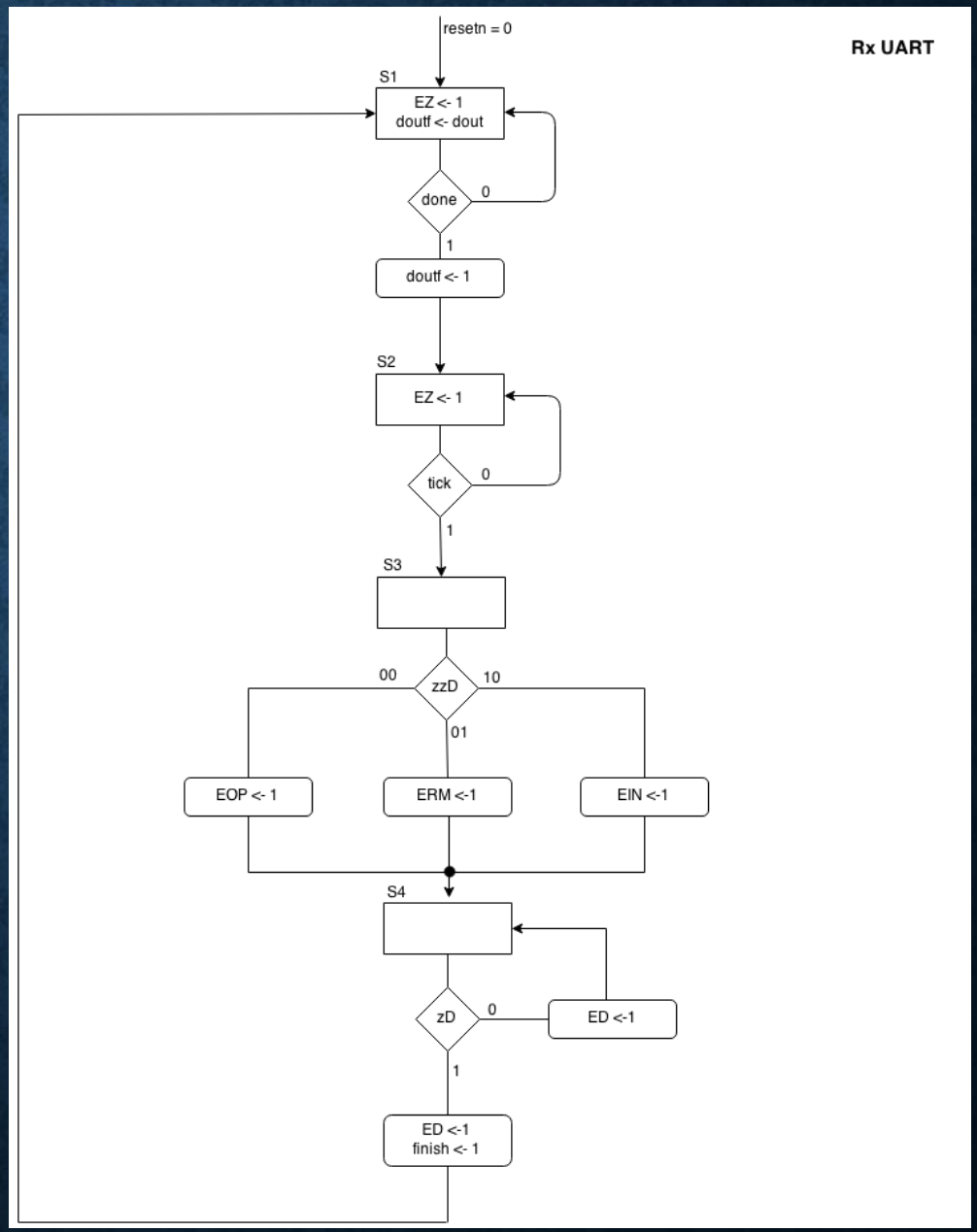
Extracting portion of bits

ASM CHART

RX receiver component



RX top



ASCII TO INSTRUCTION CODE

op code (select)	ASC2	HEX
00000'	@	40
00001'	A	41
00010'	B	42
00011'	C	43
00100'	D	44
00101'	E	45
00110'	F	46
00111'	G	47
01000'	H	48
01001'	I	49
01010'	J	4A
01011'	K	4B
01100'	L	4C
01101'	M	4D
01110'	N	4E
01111'	O	4F
10000'	P	50
10001'	Q	51
10010'	R	52
10011'	S	53
10100'	T	54
10101'	U	55
10110'	V	56
10111'	W	57
11000'	X	58
11001'	Y	59
11010'	Z	5A
11011'	[	5B
11100'	\	5C
11101'	]	5D
11110'	^	5E
11111'	_	5F

K = 01001011

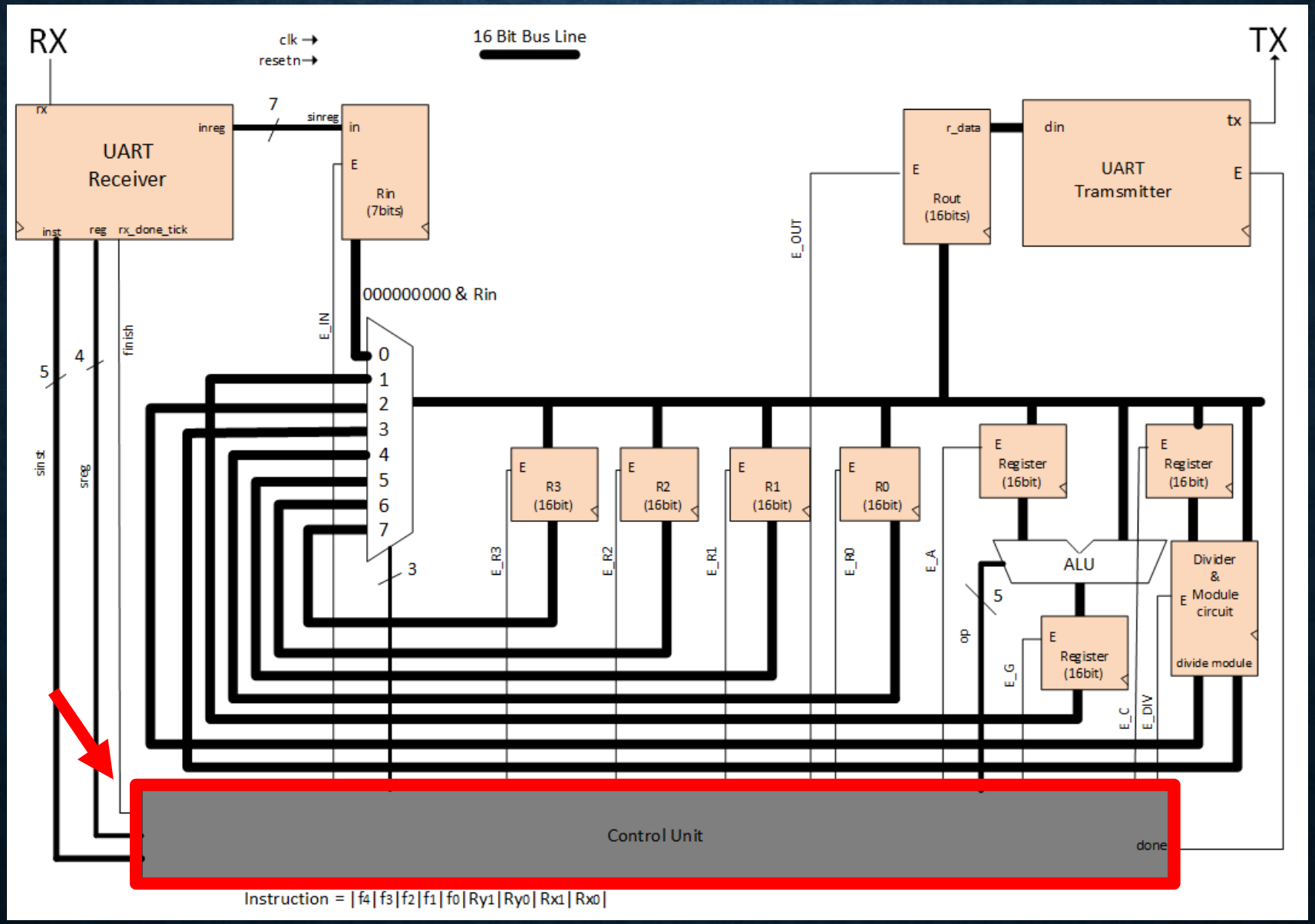
Register Selection		BINARY	ASC2	HEX
R0	R0	0000`	@	40
R0	R1	0001`	A	41
R0	R2	0010`	B	42
R0	R3	0011`	C	43
R1	R0	0100`	D	44
R1	R1	0101`	E	45
R1	R2	0110`	F	46
R1	R3	0111`	G	47
R2	R0	1000`	H	48
R2	R1	1001`	I	49
R2	R2	1010`	J	4A
R2	R3	1011`	K	4B
R3	R0	1100`	L	4C
R3	R1	1101`	M	4D
R3	R2	1110`	N	4E
R3	R3	1111`	O	4F
Taking last 4 bits				

D = 01000100

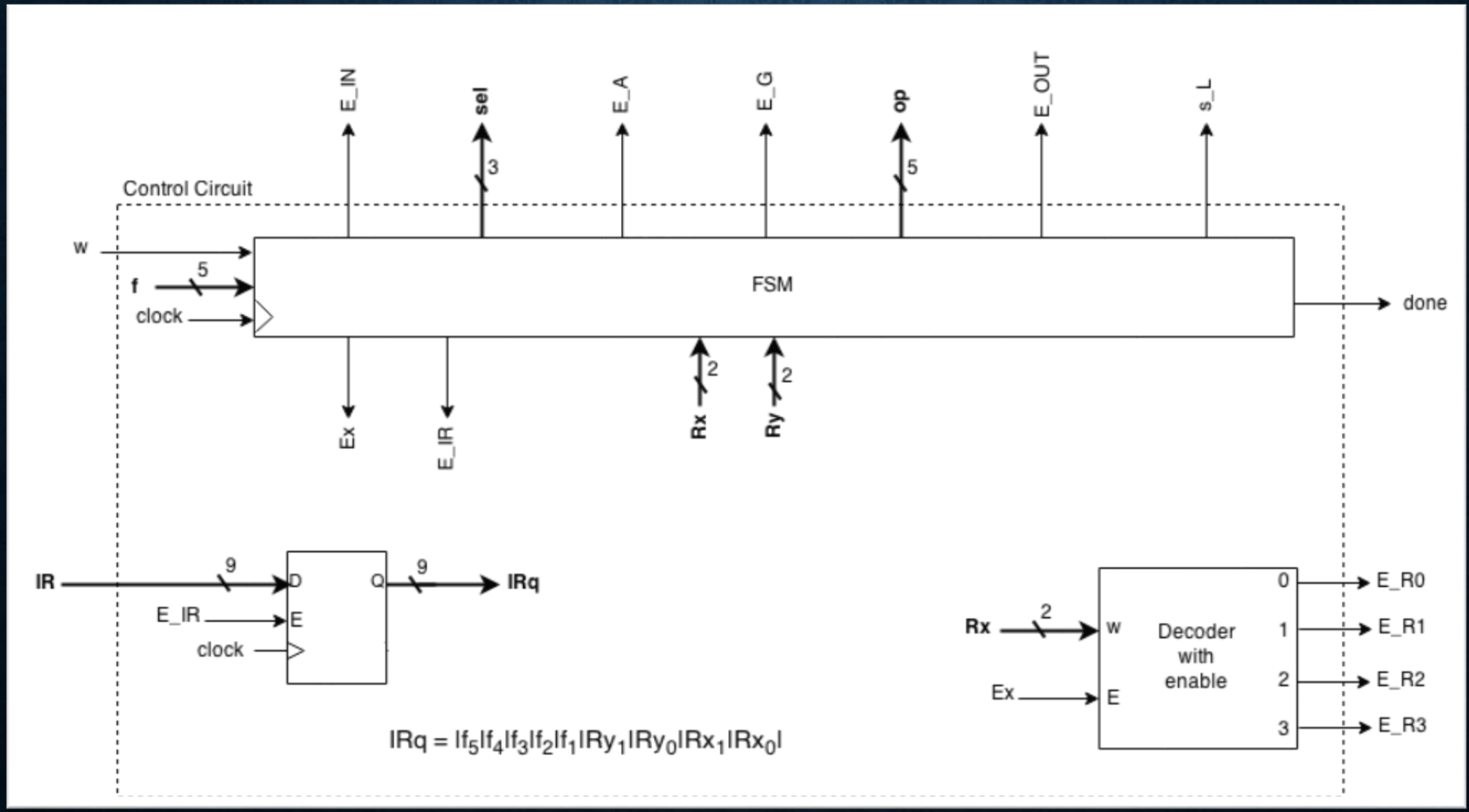
For input 7 bit numbers

1. Convert to decimal number
2. Sending "Chr(decimal number)
3. Taking last 7 bit at board

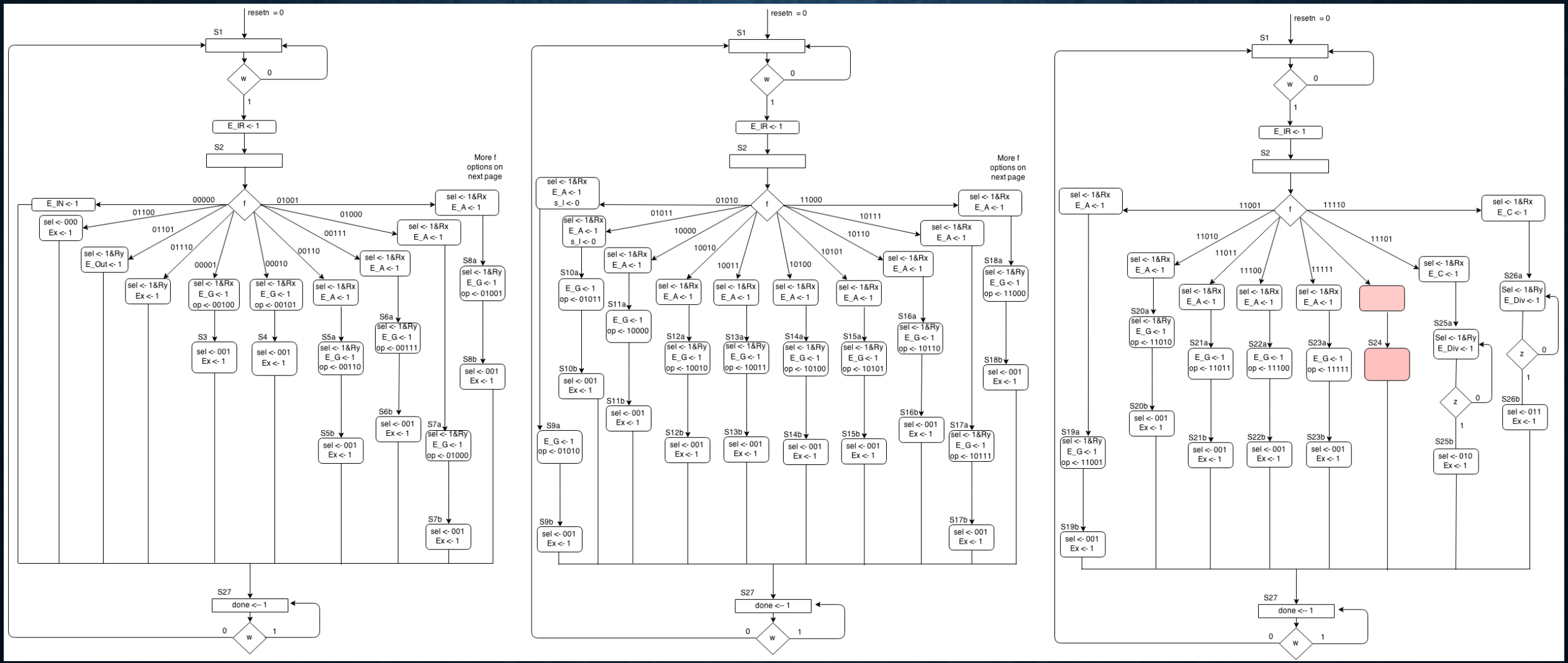
TOP LEVEL DESIGN



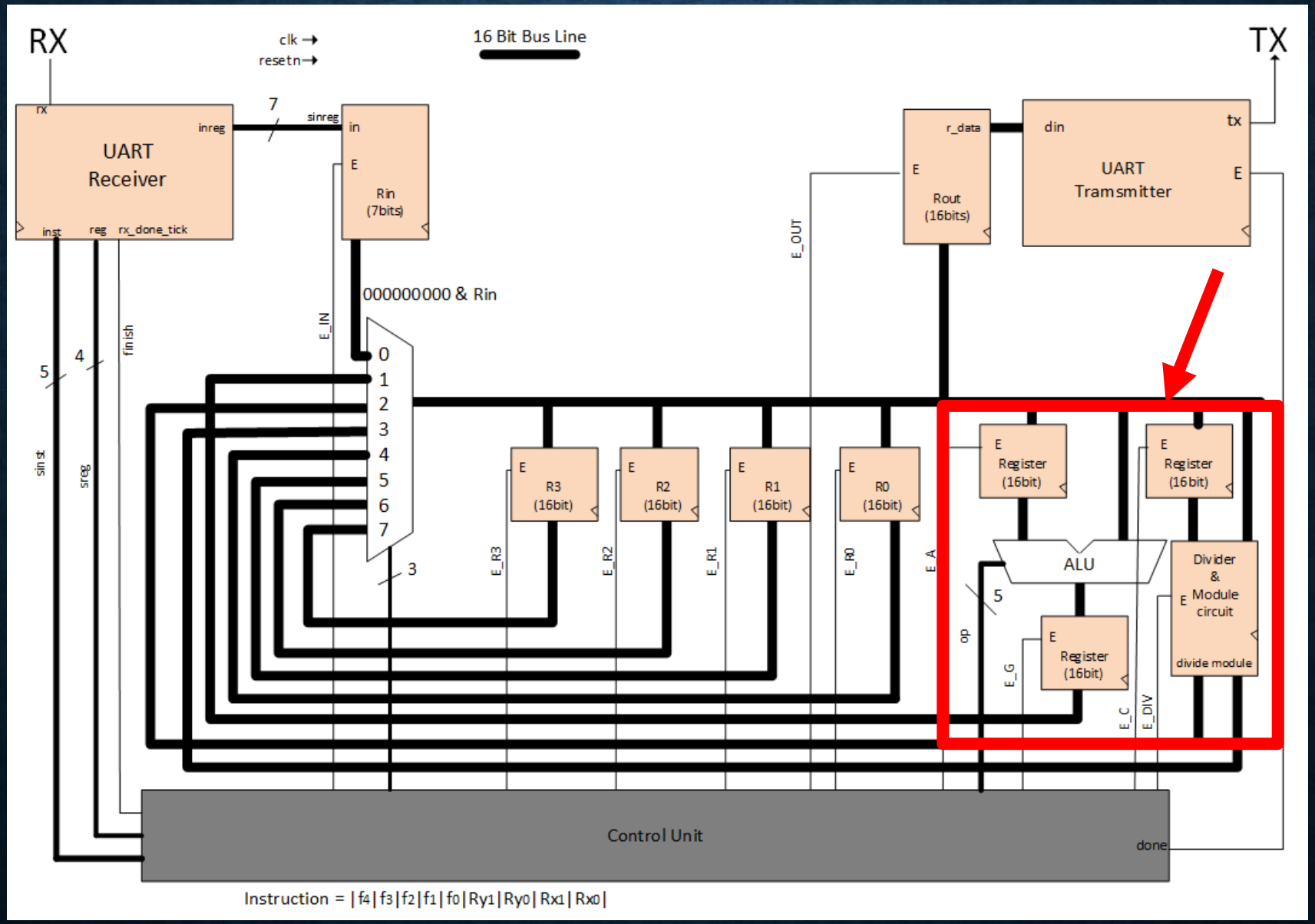
CONTROL UNIT



CONTROL UNIT ASM CHART



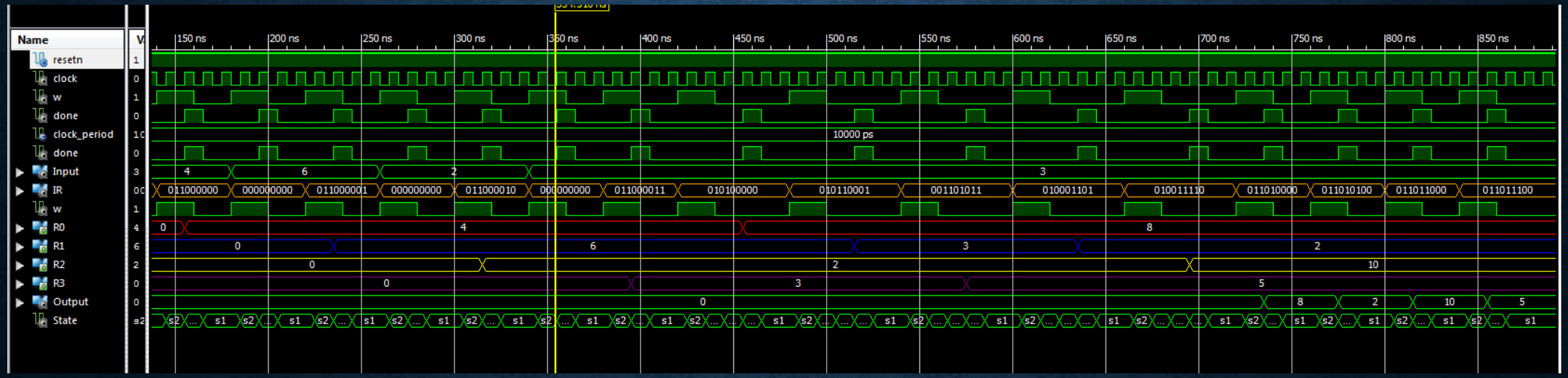
TOP LEVEL DESIGN



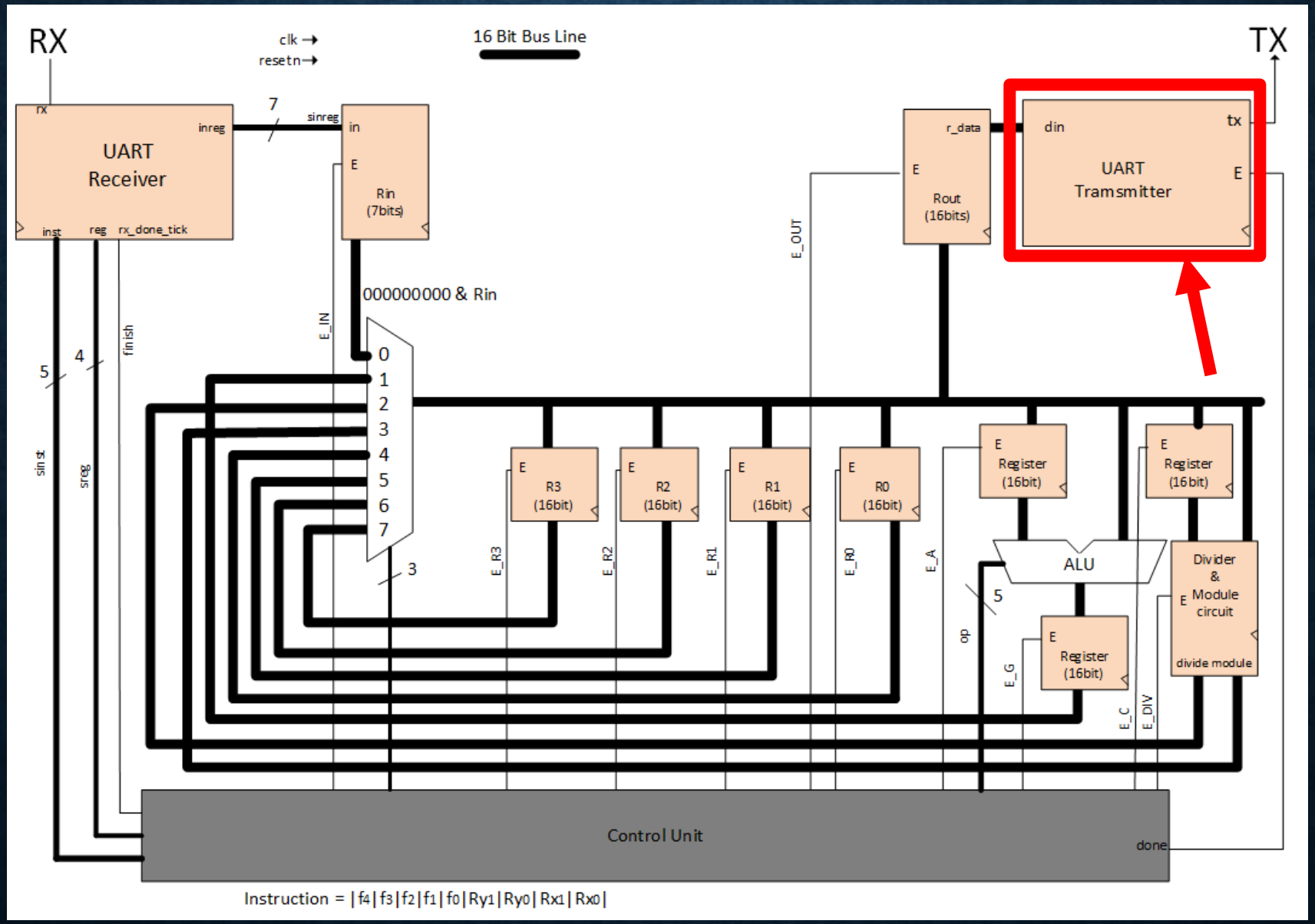
ALU INSTRUCTION

00000'	load, IN	IN<-User	S2
00001'	increment	RX<-RX+1	S3
00010'	decrement	RX<RX-1	S4
00011'	B	???	????
00100'	B+1	???	????
00101'	B-1	???	????
00110'	add	RX<-RX+RY	S5a,S5b
00111'	subtract	RX<-RX-RY	S6a,S6b
01000'	absolute subtration	RX<-|RX-RY|	S7a,S7b
01001'	multiply	RX<-RX*RY	S8a.S8b
01010'	left shift	RX<->RX	S9a,S9b
01011'	right shift	RX<-<RX	S10a,S10b
01100'	load Rx, IN	Rx<- IN	S2
01101'	load out	Out<-RY	S2
01110'	copy	RX<-RY	S2
01111'			
10000'	not(A)	RX<-Not(RX)	S11a,S11b
10001'	not(B)	????	????
10010'	AND	RX<-RX AND RY	S12a,S12b
10011'	OR	RX<-RX OR RY	S13a,S13b
10100'	NAND	RX<-RX NAND RY	S14a,S14b
10101'	NOR	RX <- RX NOR RY	S15a,S15b
10110'	XOR	RX<- RX XOR	S16a,S16b
10111'	XNOR	RX<- RX XNOR RY	S17a,S17b
11000'	Greater than (output greater input	RX <- larger of RX, RY	S18a,S18b
11001'	Less than (output less input)	RX<- smaller of RX and RY	S19a,S19b
11010'	Equal to (output input)	RX<-RX if Equal	S20a,S20b
11011'	Binary	RX<-Gray to Binary(RX)	S21a,S21b
11100'	Gray	RX<-Binary to Gray(RX)	S22a,S22b
11101	Divide Rx by RY	RX<- RX/RY	S25a,S25b
11110	Rx mod Ry	RX<-RX mod RY	S26a,26b
11111	Reset	RX<-"00000000"	S23a,S23b

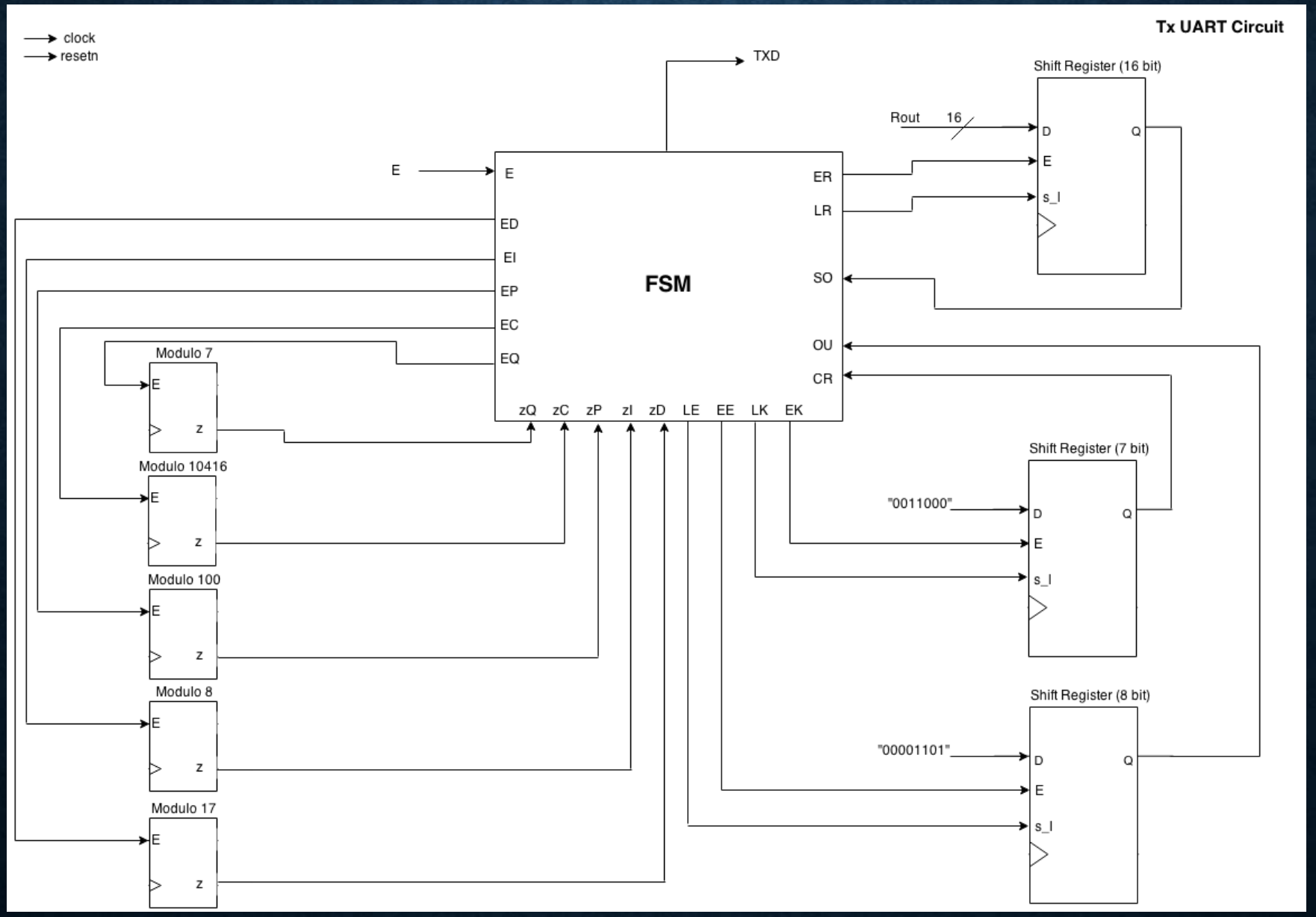
TEST BENCH


Name
resetn
clock
w
done
clock_period
done
Input
IR
w
R0
R1
R2
R3
Output
State
150 ns
200 ns
250 ns
300 ns
350 ns
400 ns
450 ns
500 ns
550 ns
600 ns
650 ns
700 ns
750 ns
800 ns
850 ns
10000 ps
011000000
000000000
011000001
000000000
011000010
000000000
011000011
010100000
010110001
001101011
010001101
010011110
011010000
011010100
011011000
011011100
s1
s2

TOP LEVEL DESIGN



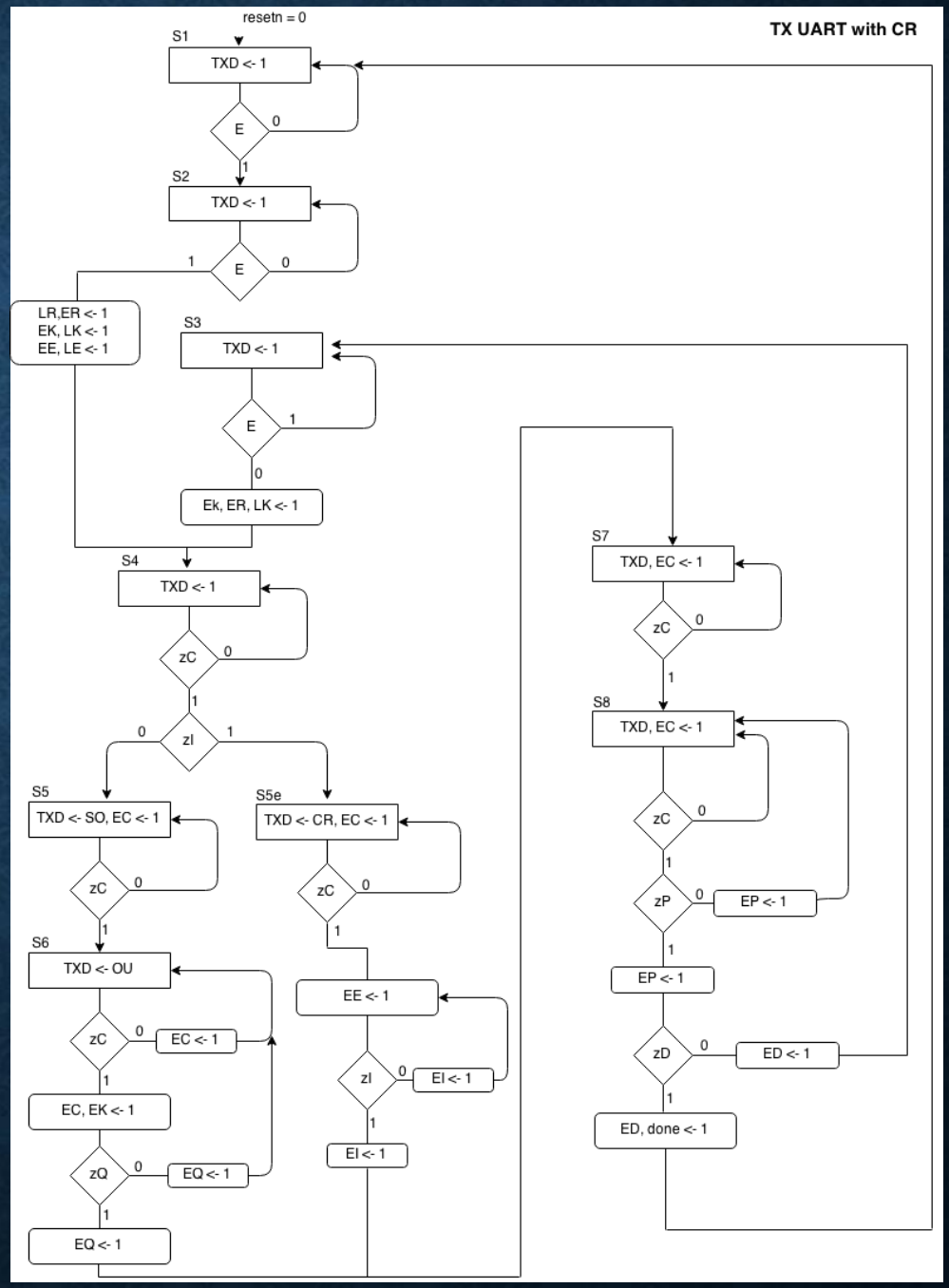
UART (TX) DATA PATH



ASM CHART

Adding 0011000 to convert to ASCII encoding

Sending Carriage Return at the end



LIVE DEMONSTRATION