Arithmetic Logic Unit and Conditional Jump

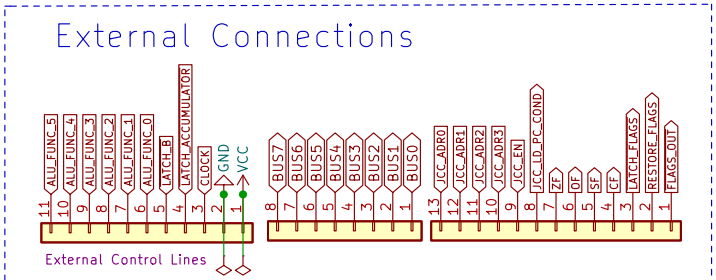
TRUTH TABLE

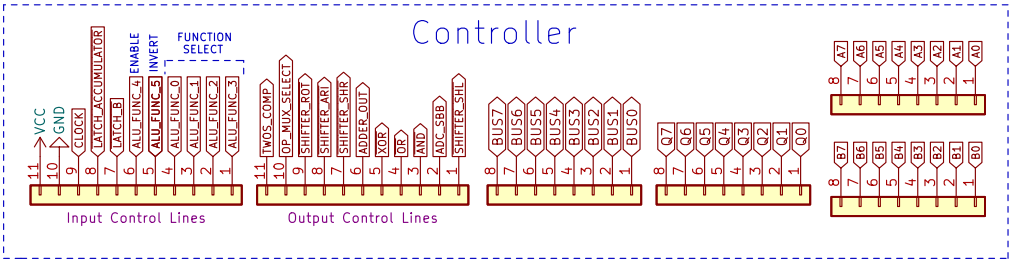
		A*	ADD	ADC	SUB	SBB	INC	DEC	AND	OR	XOR	SHL	SHR	ASL	ASR	ROR	ROL	CMP	TEST	NAND	NOR	XNOR	NOT A
AI	L	_/	_/	_/	_/	_/	_/	_/	_/	_/	_/	_/	_/	_/	_/	_/	_/	_/	_/	_/	_/	_/	_/
BI	L	_/	_/	_/	_/	_/	_/	_/	_/	_/	_/	_/	_/	_/	_/	_/	_/	_/	_/	_/	_/	_/	_/
FLAG_IN	L	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0
ALU F0		0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	1	1	1	0	1	0
ALU F1		0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	1	1	1	0	0	0
ALU F2		0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	1	1	0	0	0
ALU F3		0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	1	1	0
(Invert Output) F4		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0
(Enable ALU) F5		1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

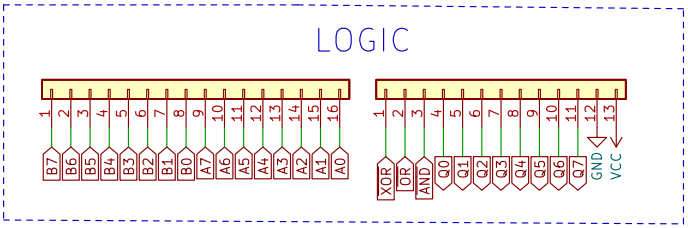
		NOP	JP	JLE JNG	JG JNLE	JGE JNL	JL JNGE	JA JNBE	JBE JNA	JNB JAE JNC	JB JNAE JC	JNE JNZ	JE JZ	JNS	JS	JNO	JO
JCC_EN	L	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
JCC_ADD3		x	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
JCC_ADD2		x	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1
JCC_ADD1		x	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1
JCC_ADD0		x	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1

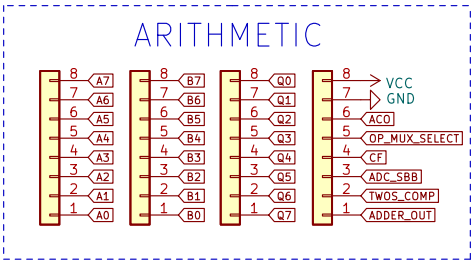
		PUSH FLG*	PUSH FLG*
FLAG_OUT	L	L	L
RESTORE_FLAGS	H	L	H

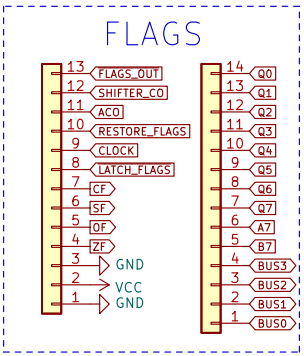
ALU Backplane

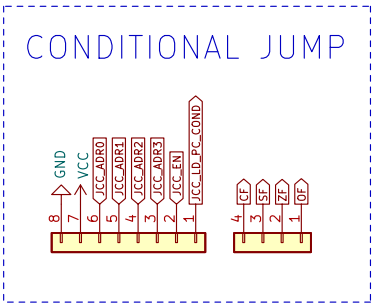


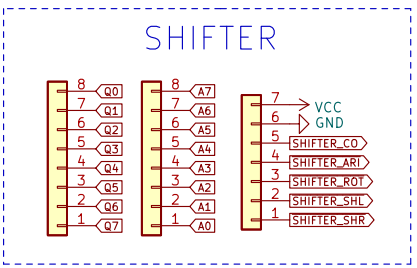












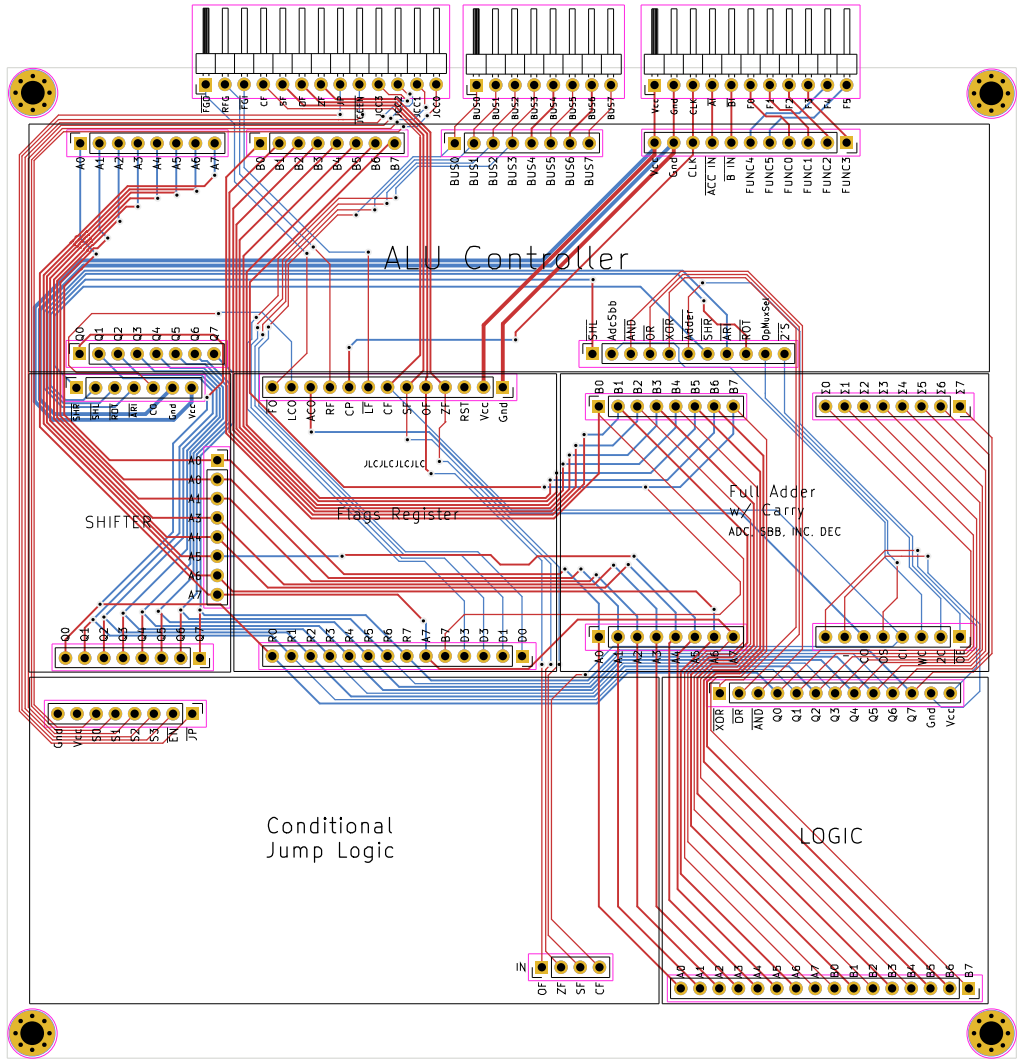
theWickedWebDev/8-bit-computer

Sheet: /

File: ALU-Backplane.kicad_sch

Title: ALU & Conditional Jump Backplane

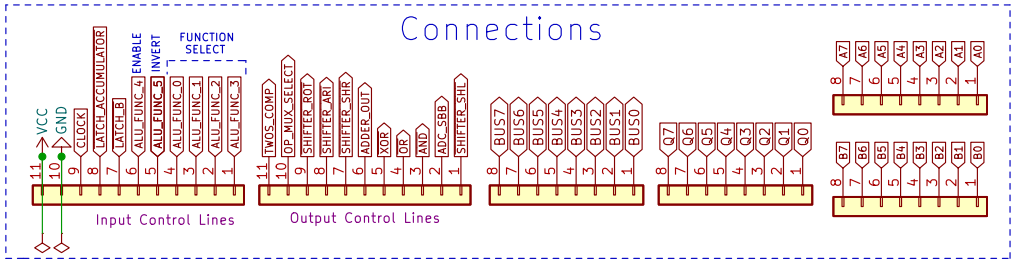
Size: User	Date: 2022-01-17	**Rev: 4**
KiCad E.D.A. kicad (6.0.0-0)		Id: 1/1


ALU Controller
SHIFTER
Flags Register
Full Adder
w/ Carry
ADC, SBB, INC, DEC
Conditional
Jump Logic
LOGIC

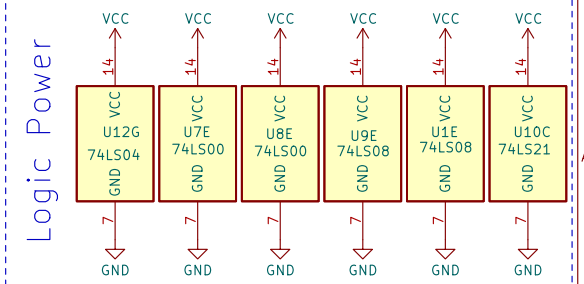
ALU Control Module

A + 1	AND	SHL
A − 1	NAND	SHR
A + B	OR	ASL
A − B	NOR	ASR
A	XOR	ROR
NOT A	XNOR	ROL

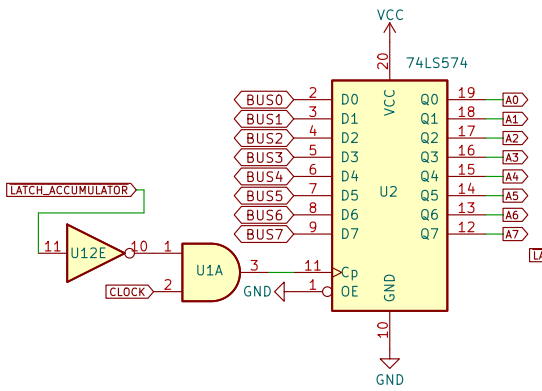
Connections



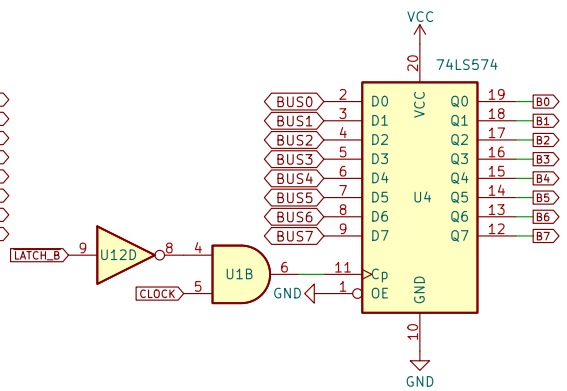
Logic Power



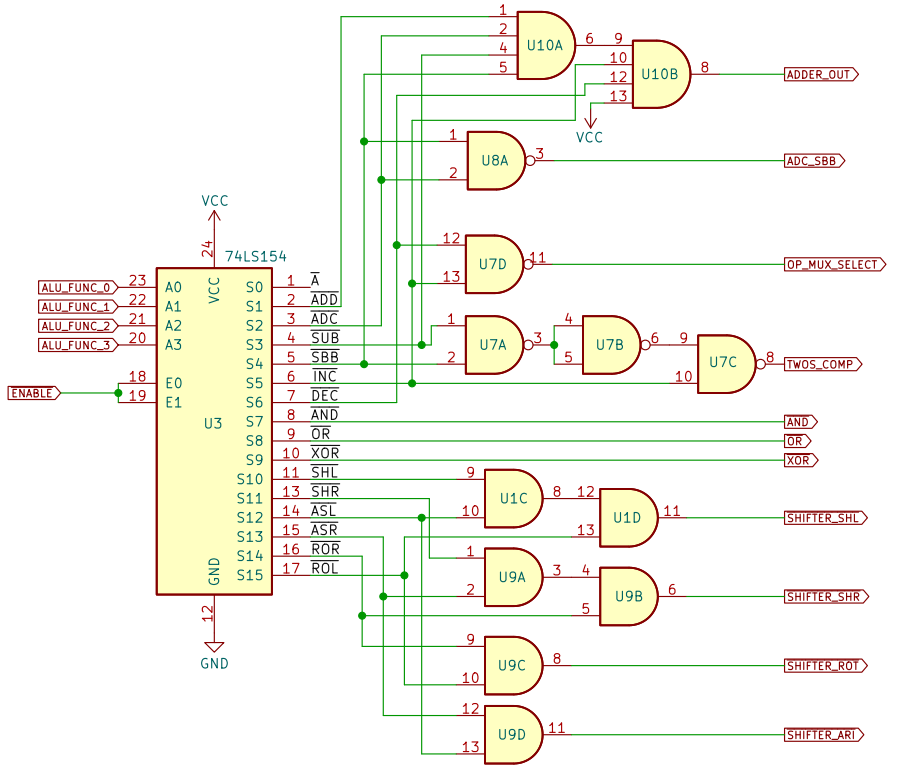
ACCUMULATOR (A)



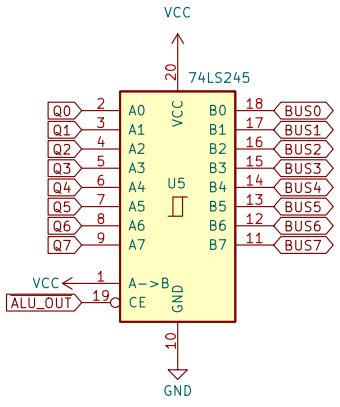
OPERAND (B)



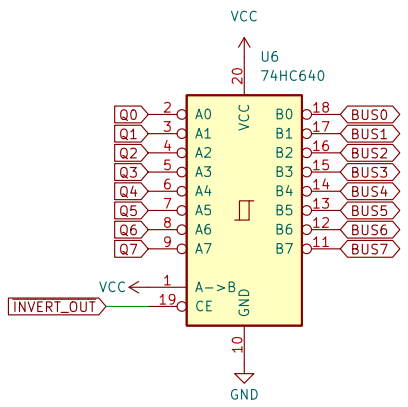
FUNCTION MULTIPLEXER



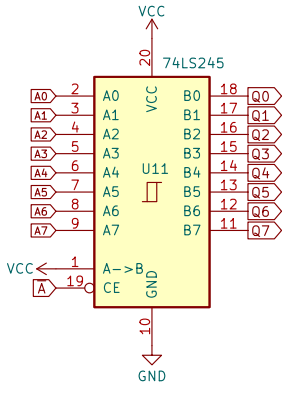
ALU OUTPUT



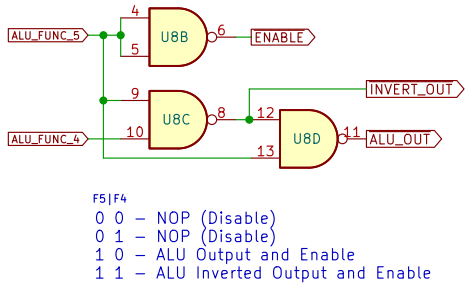
INVERTED ALU OUTPUT



PASS A



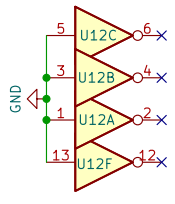
Enable and Invert Control



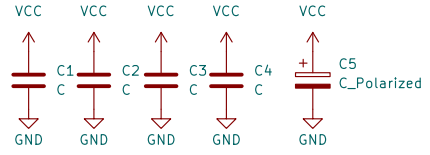
F5|F4

0 0 – NOP (Disable)
0 1 – NOP (Disable)
1 0 – ALU Output and Enable
1 1 – ALU Inverted Output and Enable

Unused



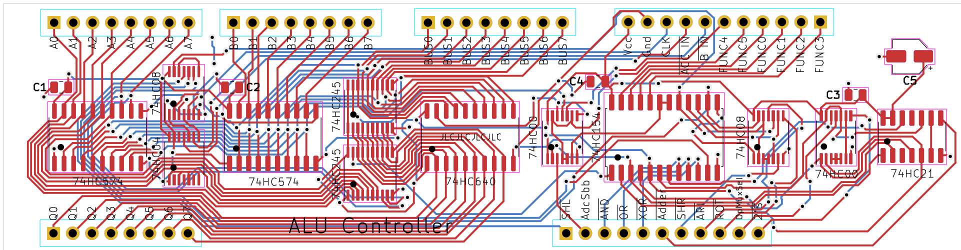
Smoothing Caps



Sheet: /
File: Control_and_Output.kicad_sch

Title:

Size: User | Date: | **Rev:**
KiCad E.D.A. kicad (6.0.0-0) | Id: 1/1



ALU Controller
A0
A1
A2
A3
A4
A5
A6
A7
B0
B1
B2
B3
B4
B5
B6
B7
Vcc
Gnd
CLK
FUNC4
FUNC5
FUNC0
FUNC1
FUNC2
FUNC3
Q0
Q1
Q2
Q3
Q4
Q5
Q6
SHL
AdcSbb
AND
OR
XOR
AdBF
SHR
AR
RO
C1
C2
C3
C4
C5
74HC574
74HC245
74HC640
74HC08
74HC00
74HC21
JLCJLCJLCJLC

Arithmetic Module

8bit Arithmetic Module provides
ADD, ADC, SUB, SBB, INC, DEC

DEC A: MUX_SEL: 1, TC: 0, CI: 0

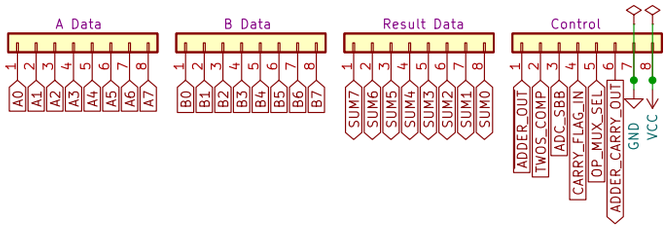
A − B: MUX_SEL: 0, TC: 1, CI: 1

A + B: MUX_SEL: 0, TC: 0, CI: 0

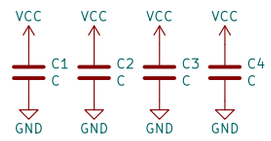
INC A: MUX_SEL: 1, TC: 1, CI: 1

A − B − Ci: MUX_SEL: 0, TC: 1, CI: ?
A + B + Cf: MUX_SEL: 0, TC: 0, CI: ?

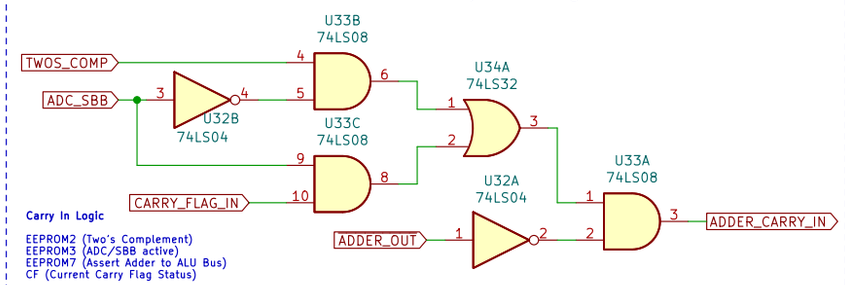
Connectors



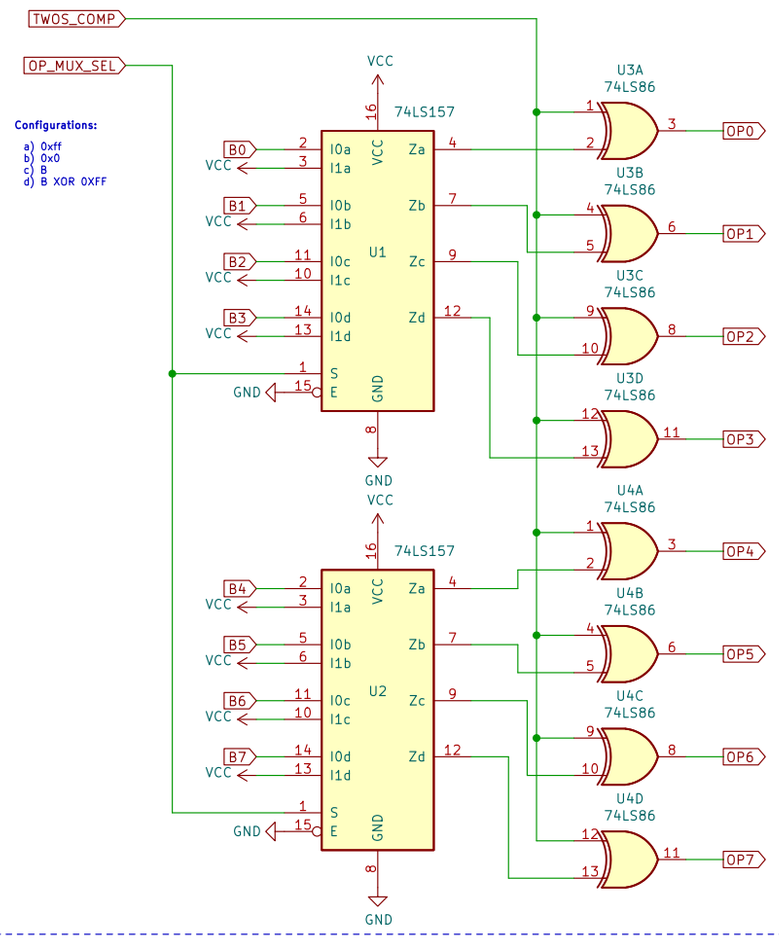
Smoothing Caps



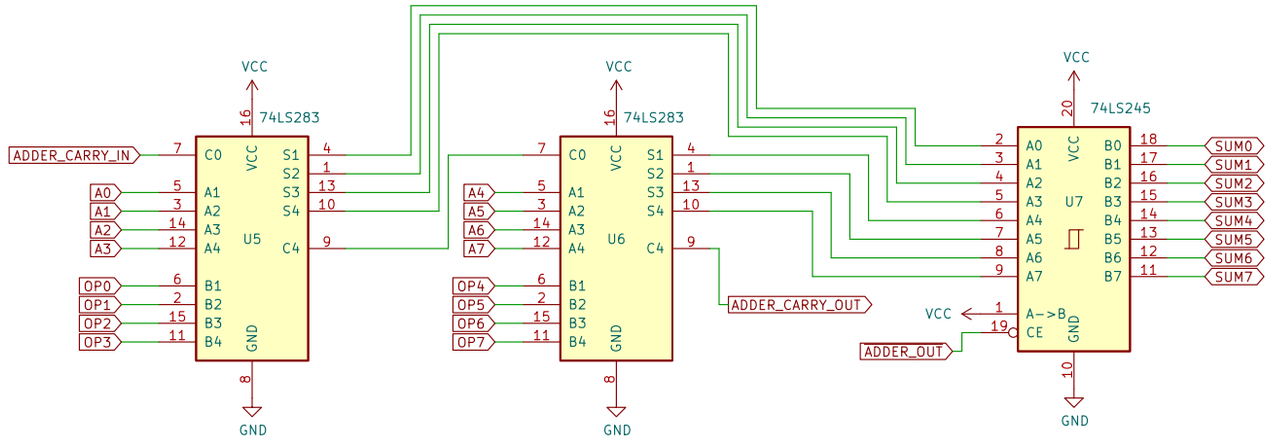
Arithmetic Carry In Logic



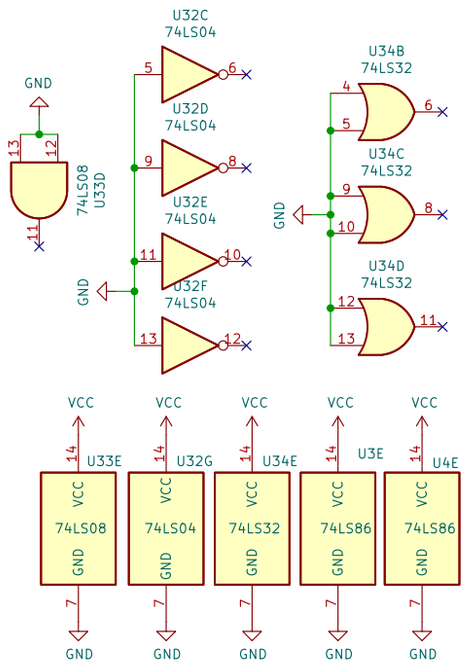
Operand Multiplexer



Full Adder w/ Carry



Logic Power

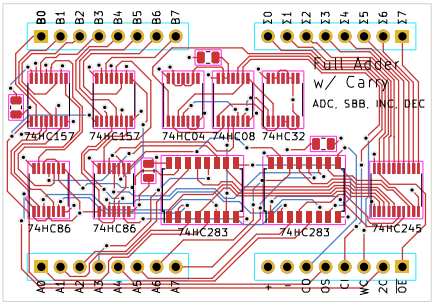


ADD / SUB / ADC / SBB / INC / DEC

Sheet: /
File: Arithmetic.kicad_sch

Title: Arithmetic Module

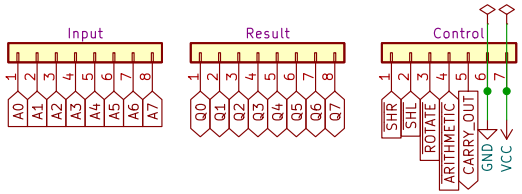
Size: User	Date:	Rev: 3
KiCad E.D.A. kicad (6.0.0−0)		Id: 1/1


B0
B1
B2
B3
B4
B5
B6
B7
Σ0
Σ1
Σ2
Σ3
Σ4
Σ5
Σ6
Σ7
Full Adder
w/ Carry
ADC, SBB, INC, DEC
74HC157
74HC157
74HC04
74HC08
74HC32
74HC86
74HC86
74HC283
74HC283
74HC245
A0
A1
A2
A3
A4
A5
A6
A7
+ / −
CO
OS
CI
WC
2C
OE

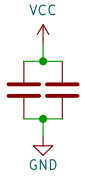
Shift & Rotate Module

Logical / Arithmetic

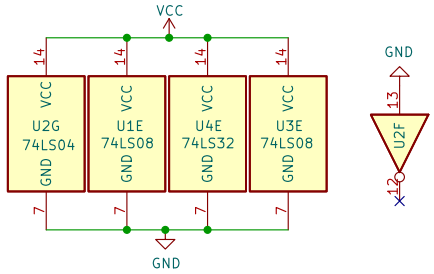
Conections



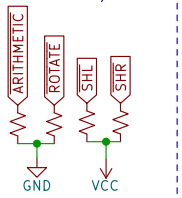
Cap's



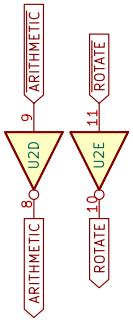
Logic Power



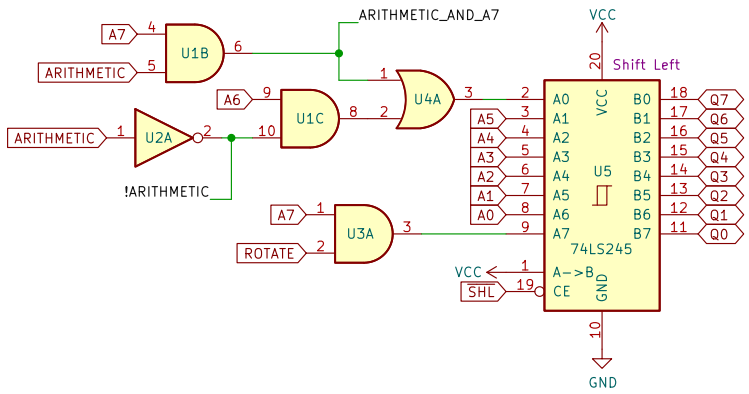
Pull U/D



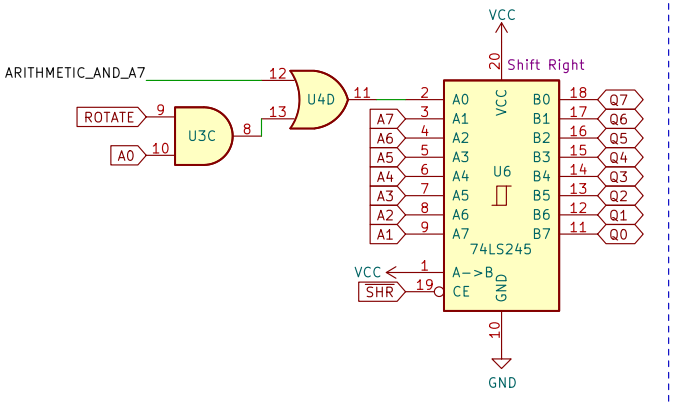
MAKE ACTIVE LOWS



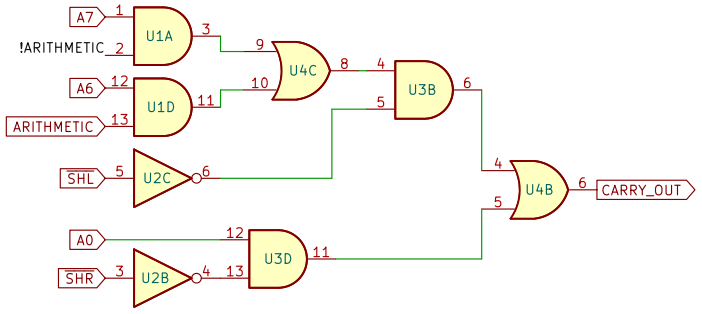
SHIFT LEFT



SHIFT RIGHT



CARRY OUT LOGIC



Rotate Enable / Carry Out

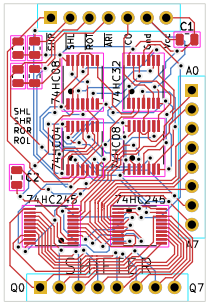
theWickedWebDev/8-bit-computer

Sheet: /

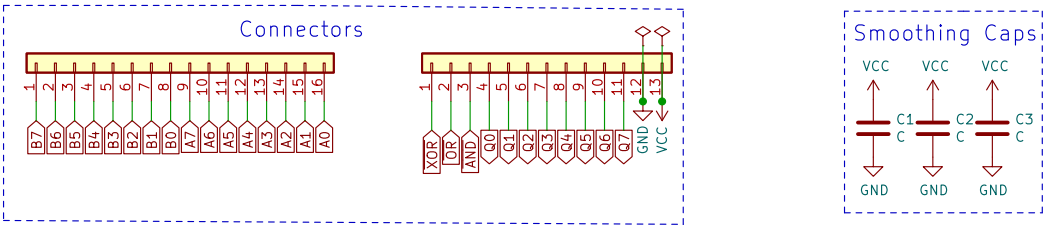
File: Shifter-v6.kicad_sch

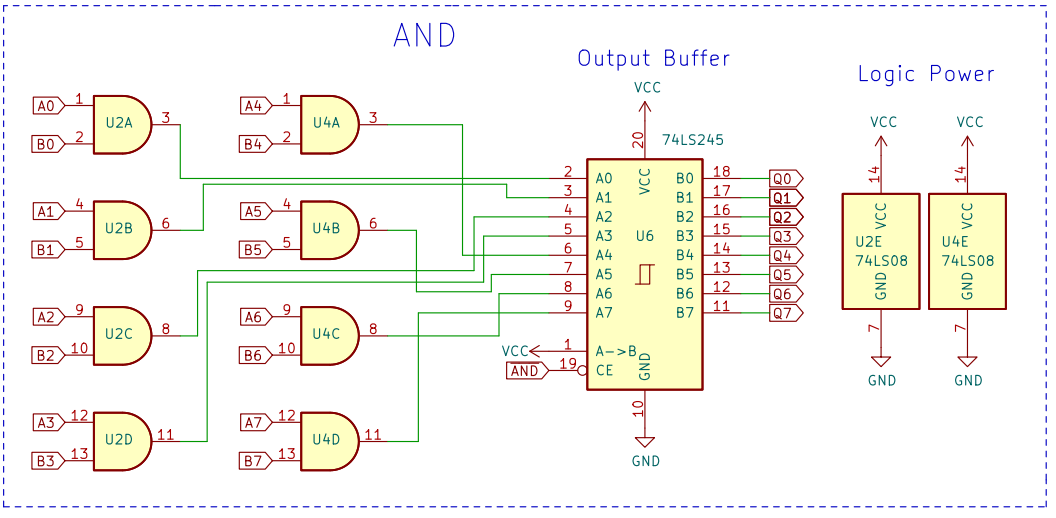
Title: Arithmetic & Logical Shifter Module

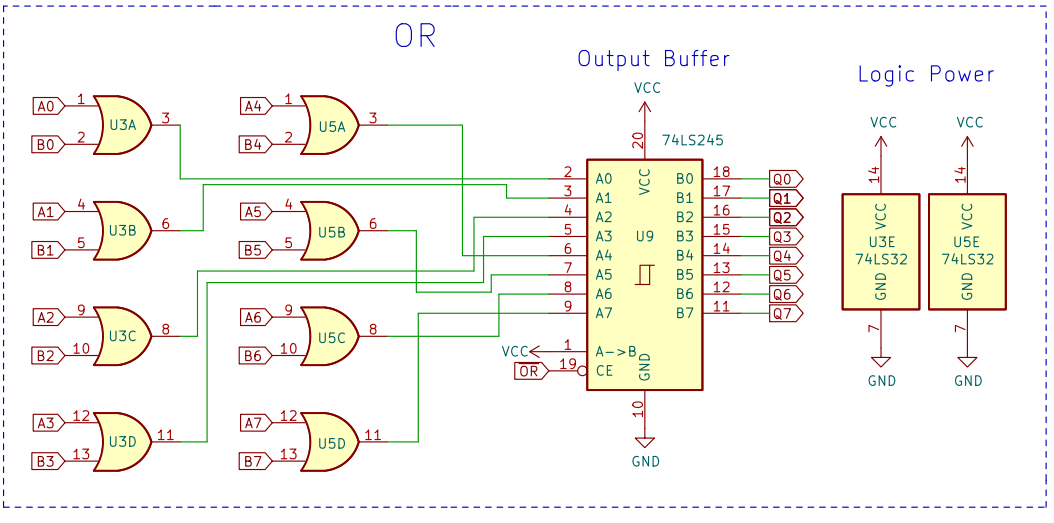
Size: A4	Date: 2022-01-16	**Rev: 4**
KiCad E.D.A. kicad (6.0.0-0)		Id: 1/1

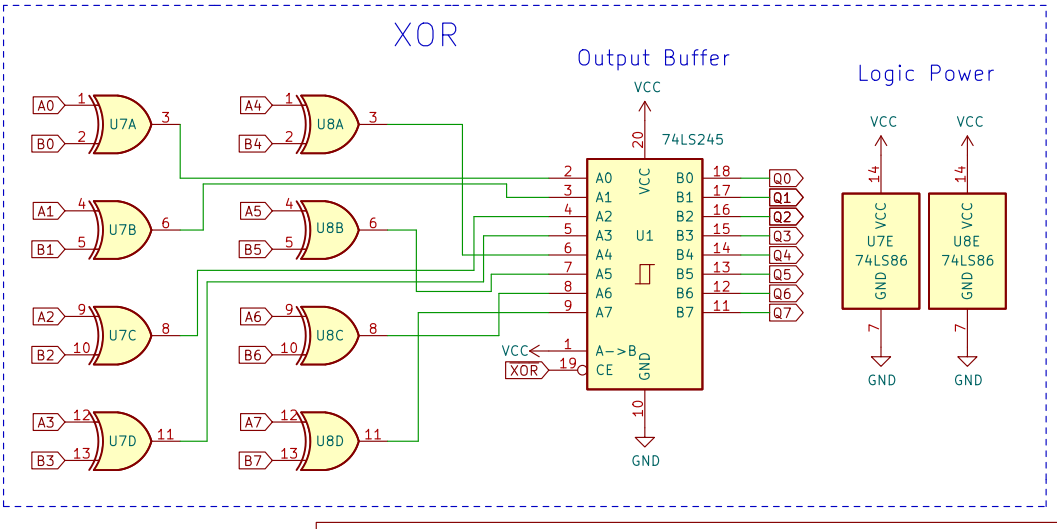

C1
A0
A7
Q0
Q7
74HC245
74HC245
74HC08
74HC32
74HC04
74HC08

8bit Logic Gate

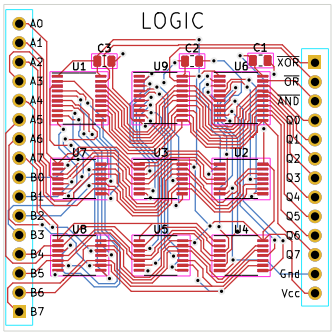








Sheet: /		
File: Modules.kicad_sch		
Title: 8bit Logic Gate (ALU)		
Size: User	Date:	**Rev: 3**
KiCad E.D.A. kicad (6.0.0-0)		Id: 1/1


LOGIC
A0
A1
A2
A3
A4
A5
A6
A7
B0
B1
B2
B3
B4
B5
B6
B7
C3
C2
C1
U1
U9
U8
U7
U3
U2
U6
U5
U4
XOR
OR
AND
Q0
Q1
Q2
Q3
Q4
Q5
Q6
Q7
Gnd
Vcc

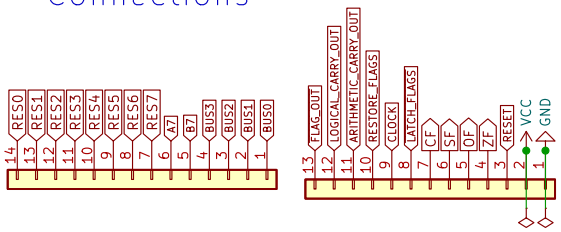
FLAGS REGISTER

$\overline{\text{LATCH\_FLAGS}}$ – A LOW signal will store the data asserted from the multiplexer into the Flags Register (FR)

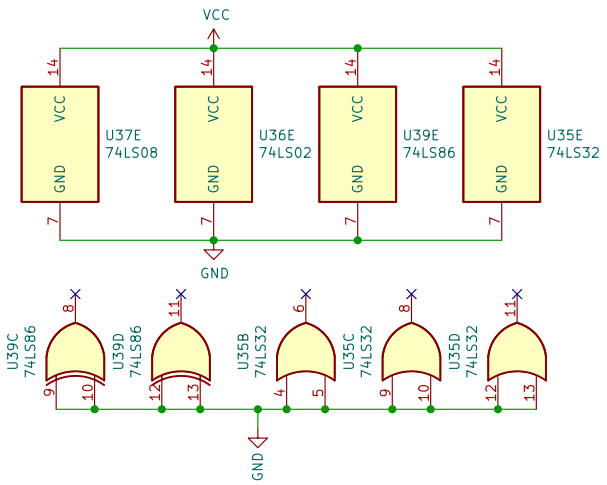
- $\overline{\text{RESTORE}}$: LOW, uses signals from ALU
- RESTORE: HIGH, uses signal asserted on data bus

$\overline{\text{FLAG\_OUT}}$ – Assers the current flags statuses onto the Data bus, typically used to push it onto the stack to handle an ISR

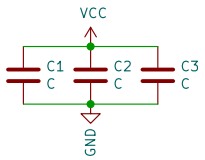
Connections



Logic Gate Power

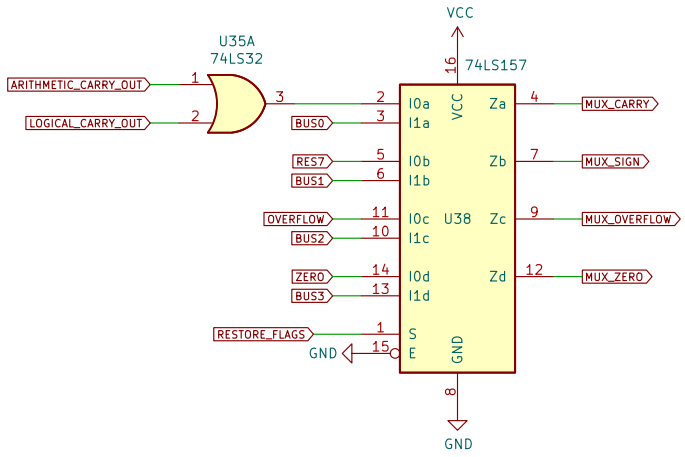


Smoothing Caps

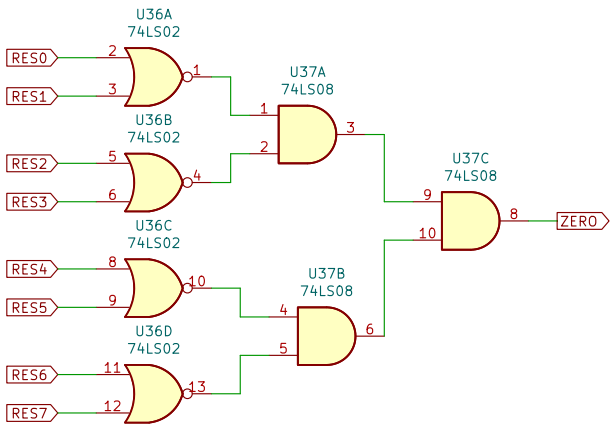


Source Multiplexer

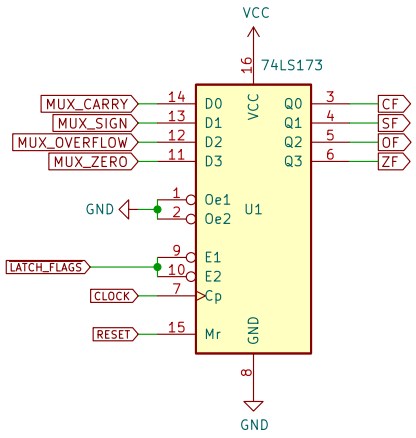
Flags come directly from ALU, or, from the flag/data bus to restore flags from the stack or another location



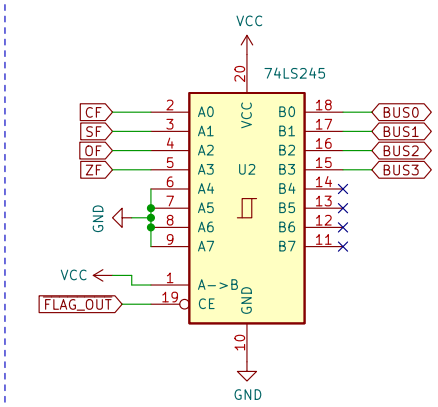
Zero



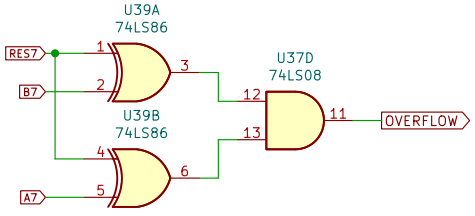
REGISTER



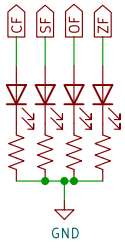
Bus Connection



OVERFLOW



Leds



For storing and asserting current flag statuses from ALU

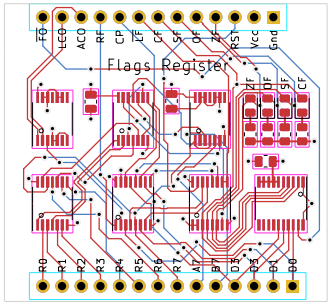
theWickedWebDev/8-Bit-Computer

Sheet: /

File: Flags Register.kicad_sch

Title: Flags Register

Size: User	Date: 2022-01-03	**Rev: 3**
KiCad E.D.A. kicad (6.0.0-0)		Id: 1/1


Flags Register
F0
LEO
ACO
RF
CP
ZF
CF
SF
OF
IF
RST
Vcc
Gnd
ZF
OF
SF
CF
R0
R1
R2
R3
R4
R5
R6
R7
A7
B7
D3
D5
D1
D0