ECE 383 - Embedded Computer Systems II

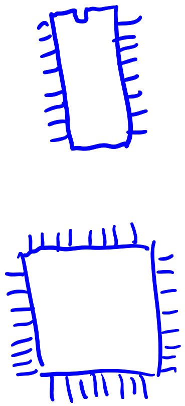
Lecture 3 - Combinational Element, unsigned, constraints file, synthesis

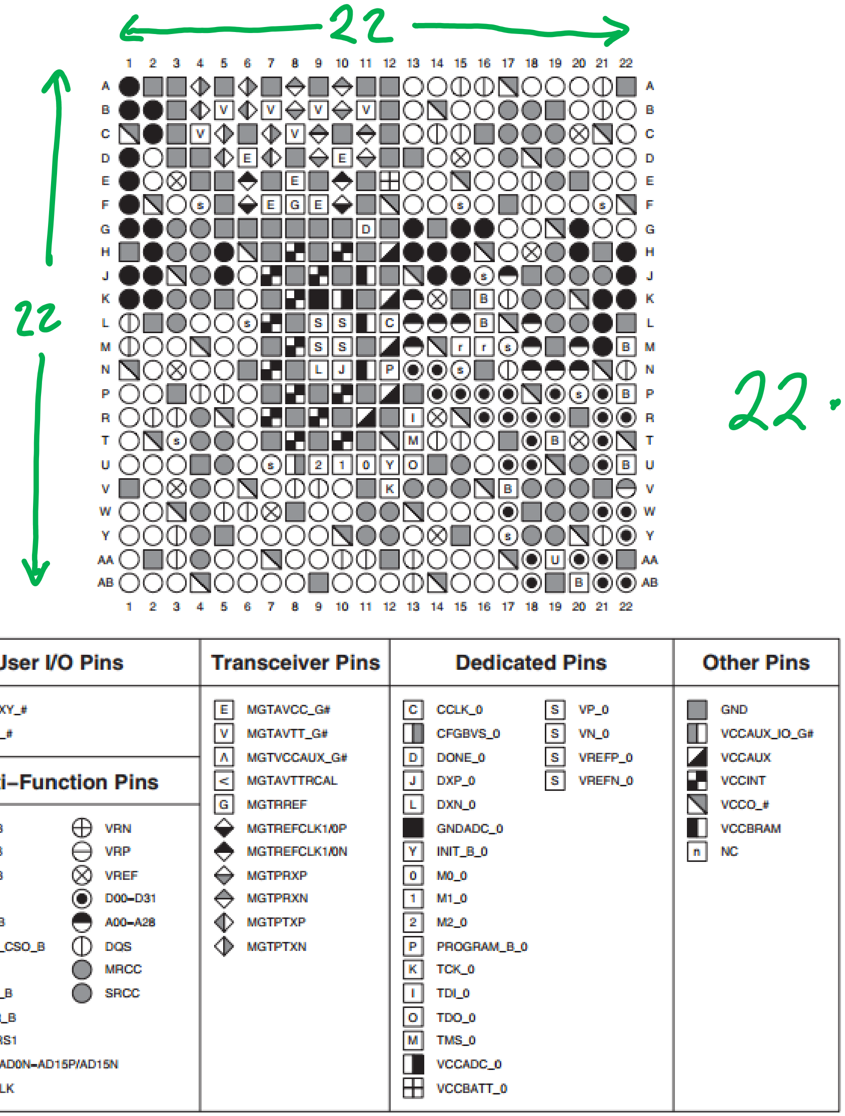
UNITED STATES AIR FORCE ACADEMY

Lesson Outline

1. **Synthesis**
2. **Constraints file**
3. **Combinational Element**
4. **Unsigned Numeric Standard**
5. **Combinations**

SB484, SBG484, SBV484, and RS484 Packages—XC7A200T





ug475_c3_309_070612

pp 80

$22 \cdot 22 = 484$ pins

Figure 3-29: **SB484, SBG484, SBV484, and RS484 Packages—XC7A200T Pinout Diagram**

https://www.xilinx.com/support/documentation/user_guides/ug475_7Series_Pkg_Pinout.pdf

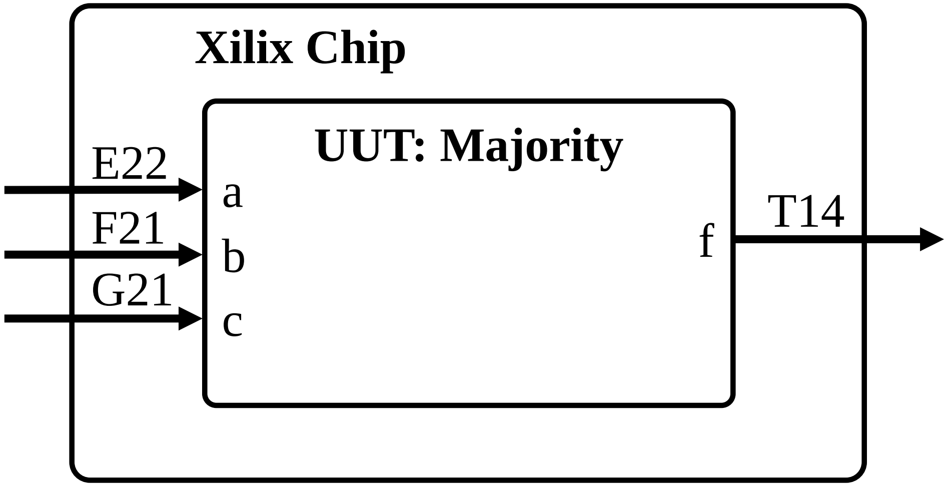
Synthesis

Synthesis

lec01.xdc

- Insert this code into your Majority.xdc file
  - Inputs from switches and outputs to LEDs

```
# This is slide switch SW0
set_property -dict { PACKAGE_PIN E22  IOSTANDARD LVCMOS12 } [get_ports { a }]; #IO_L22P_T3_16 Sch=sw[0]
# This is slide switch SW1
set_property -dict { PACKAGE_PIN F21  IOSTANDARD LVCMOS12 } [get_ports { b }]; #IO_25_16 Sch=sw[1]
# This is slide switch SW2
set_property -dict { PACKAGE_PIN G21  IOSTANDARD LVCMOS12 } [get_ports { c }]; #IO_L24P_T3_16 Sch=sw[2]
# This is LED Led(0)
set_property -dict { PACKAGE_PIN T14   IOSTANDARD LVCMOS25 } [get_ports { f }]; #IO_L15P_T2_DQS_13 Sch=led[0]
```



Synthesis and demo majority

Constraints file

Constraints file

- Nexyx Video Master XDC
  - ~~http://ece.ninja/383/~~ ECE383/ datasheets/NexysVideo_Master.xdc

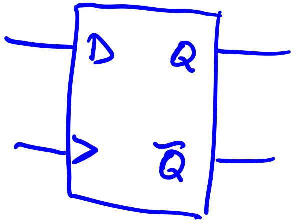
Combinational Element

Combinational Element – Common error

- Common error that may come up in your designs
- You cannot use a signal listed on the entity as an out port, on the right hand side of a signal assignment statement.

```
entity circuit is
   port (clk, data: in std_logic;
            q, not_q: out std_logic);
end circuit;

architecture error of circuit is
begin
   q <= some cool logical stuff using clk and data;
   not_q <= not q;
end error;
```



← Nope, because q is an output, not input

Combinational Element – Solution

- Solution
- assign "some cool logical stuff using clk and data" to a temporary variable

```
entity circuit is
    port (clk, data: in std_logic;
             q, not_q: out std_logic);
end circuit;

architecture error of circuit is
      signal temp std_logic;
begin
    temp <= some cool logical stuf using clk and data;
    q <= temp;
    not_q <= not temp;
end error;
```

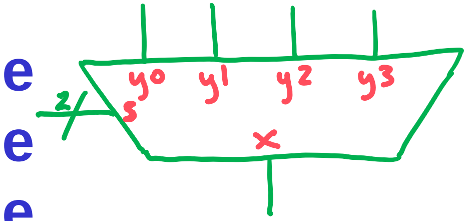
Combinational Element - Mux

Design: think in terms of B.B Bs

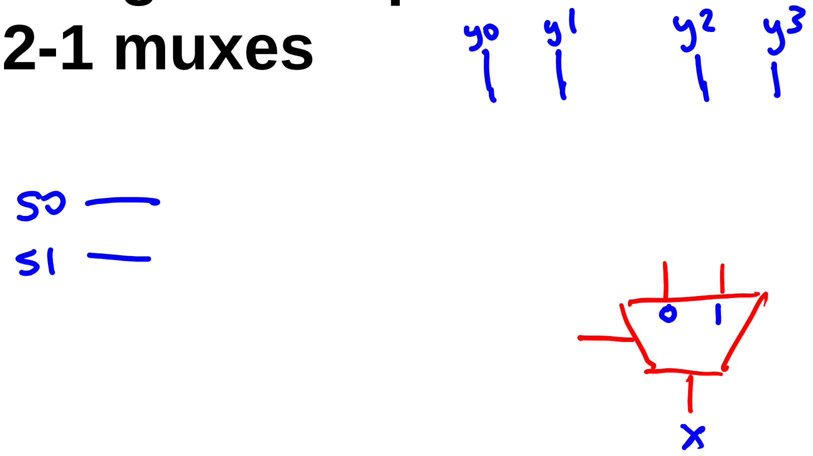
- Simplify muxes using conditional signal assignment statement
- Example:

s1 s0	X
0 0	
0 1	
1 0	
1 1	

```
x <=    y0 when S = "00" else
        y1 when S = "01" else
        y2 when S = "10" else
        y3;
```



- Draw this Circuit assuming 8-bit inputs
- Now build 4-1 mux w/ 2-1 muxes



Unsigned Numeric Standard

Unsigned Numeric Standard

- So far we mostly used STD_LOGIC_1164 library

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
```

  - Library Contents: http://www.csee.umbc.edu/portal/help/VHDL/packages/std_logic_1164.vhd

Unsigned Numeric Standard

- Numeric_Std Library supports 2 main datatypes
  - Signed and Unsigned
  - Library Contents: http://www.csee.umbc.edu/portal/help/VHDL/packages/numeric_std.vhd

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.NUMERIC_STD.ALL;
entity lec3 is
        port(       au, bu:     in unsigned(3 downto 0);
                    cu,du,su:  out unsigned(3 downto 0);
                    as, bs:      in signed(3 downto 0);
                    cs,ds,ss:  out signed(3 downto 0));
end lec3;
```

FLAGs

V -

C -

S -

Z -

Unsigned Numeric Standard

```
architecture structure of lec3 is
begin
```

unsigned

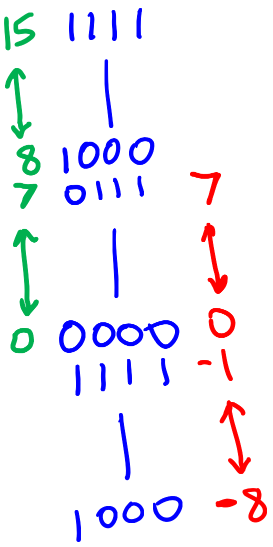
```
          cu <=        "1000" when (au > bu) else
                       "0110" when (au = bu) else
                       "0001";
          su <= au + bu;
          du <= au - bu;
```

signed

```
          cs <=        "1000" when (as > bs) else
                       "0110" when (as = bs) else
                       "0001";
          ss <= as + bs;
          ds <= as - bs;
```

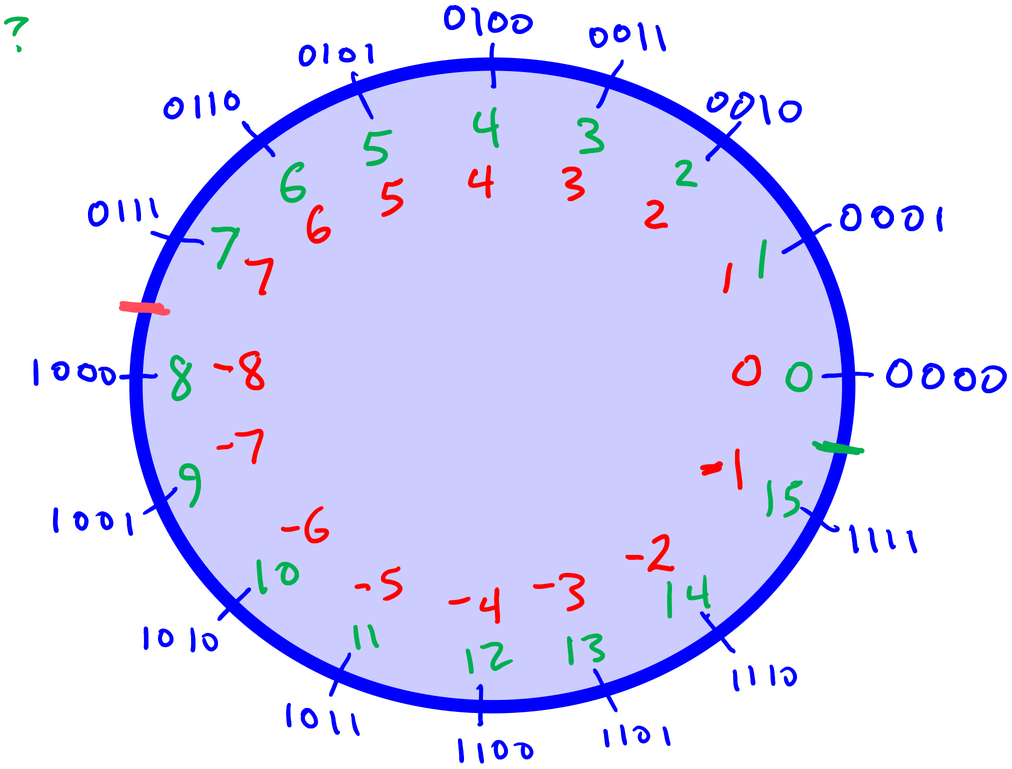
Signed vs Unsigned

- Computer/ALU has no clue if your operands are signed or unsigned
  - 2's complement math treats them the same
  - Answer is the same… only the flags(z, v, s, c) are different



GREEN?

Red?



overflow?

carry?

Unsigned Numeric Standard

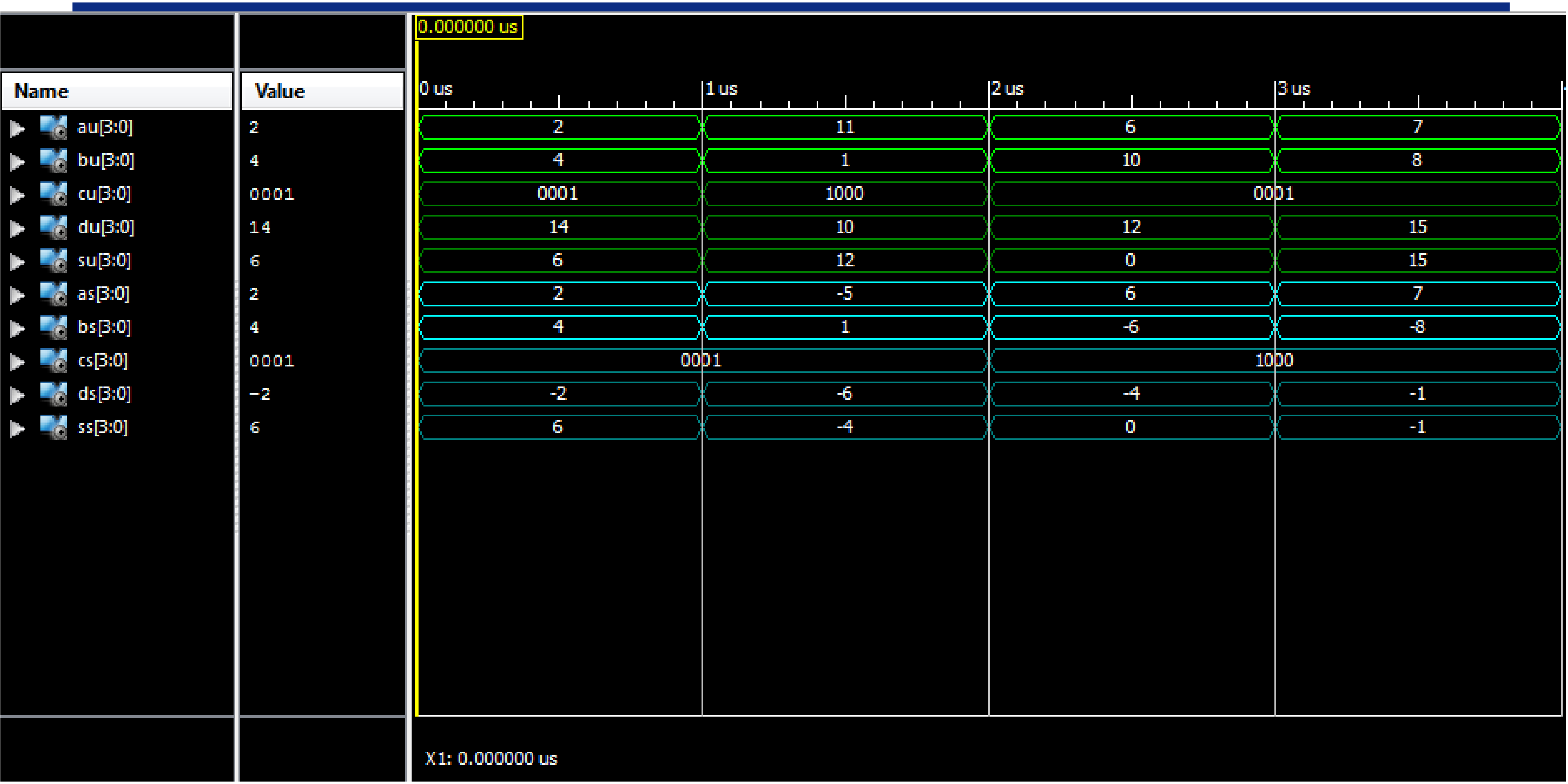
- Unsigned

A	B	Value A	Value B	A >? B	A =? B	A <? B	A + B	A - B
0010	0100							
1011	0001							
0110	1010							
0111	1000							

- Signed

A	B	Value A	Value B	A >? B	A =? B	A <? B	A + B	A - B
0010	0100							
1011	0001							
0110	1010							
0111	1000							

Unsigned Numeric Standard



0.000000 us

Name	Value	0 us	1 us	2 us	3 us
au[3:0]	2	2	11	6	7
bu[3:0]	4	4	1	10	8
cu[3:0]	0001	0001	1000	0001	
du[3:0]	14	14	10	12	15
su[3:0]	6	6	12	0	15
as[3:0]	2	2	-5	6	7
bs[3:0]	4	4	1	-6	-8
cs[3:0]	0001	0001		1000	
ds[3:0]	-2	-2	-6	-4	-1
ss[3:0]	6	6	-4	0	-1

X1: 0.000000 us

Unsigned Numeric Standard

- You will typically use STD_LOGIC_VECTOR and UNSIGNED
- You may need to convert between the two

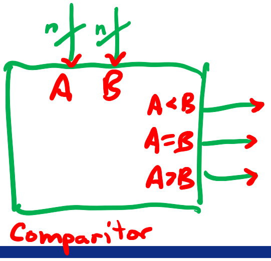
```
a: std_logic_vector(7 downto 0);
b: unsigned(7 downto 0);
c: std_logic_vector(7 downto 0);

b <= unsigned(a);
c <= std_logic_vector(b);
```

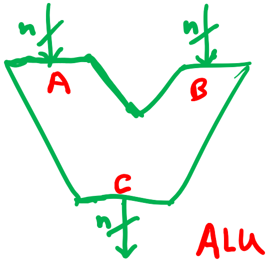
Combinations

Combinations

- Common Combinations if/then/else
- All conditional statements consist of three parts:
  - the condition to be checked (the if clause)
  - the statement to be evaluated when the condition is true (the then clause)
  - the statement to be evaluated when the condition is false (the else clause)
- Typically, the condition being evaluated seeks the relative magnitude of two unsigned binary numbers, requiring a comparator.
- The then and else clauses will typically require some logic or arithmetic operation



Comparitor



ALU

add
sub
mul divide

Combinations

- In order to illustrate the hardware realization of a conditional statement, consider the following example:

  C: if (a<4) then z=y+3 else z=y+7

  VHDL: z <= y+3 when (a < 4) else y+7;

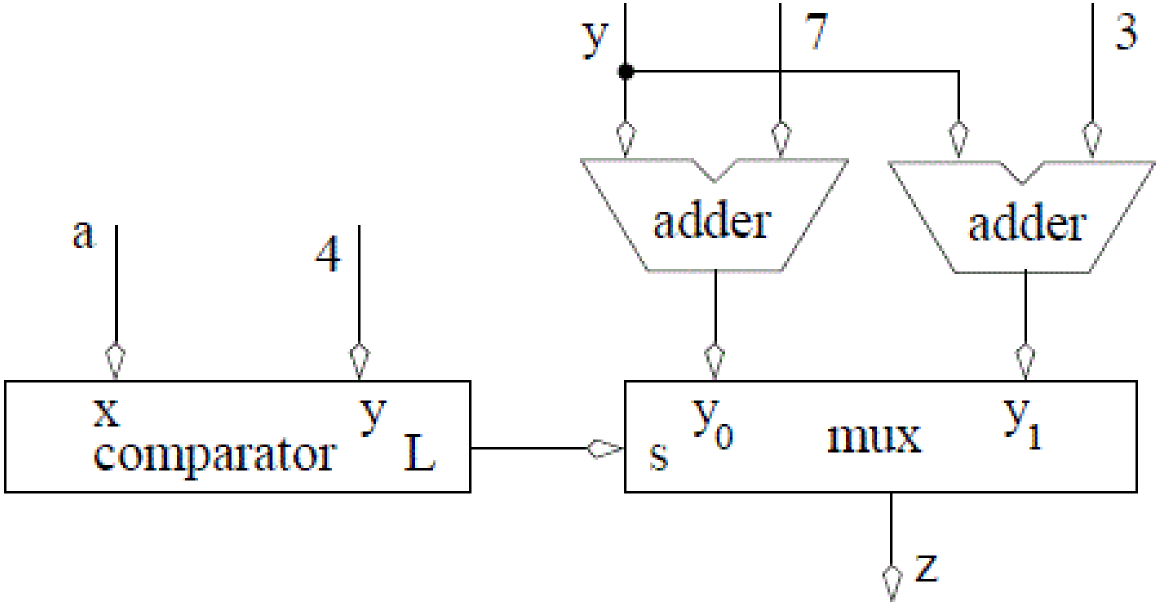
Implement with Adders, Comparitors, Muxs

Combinations

- In order to illustrate the hardware realization of a conditional statement, consider the following example:

C: if (a<4) then z=y+3 else z=y+7

VHDL: z <= y+3 when (a < 4) else y+7;



Combinations

- However, this circuit is not minimal, one of the adders can be removed.
- How?
- Practice on Homework

F = 1 if 8-bit input is divisable by 17

Do HW#3

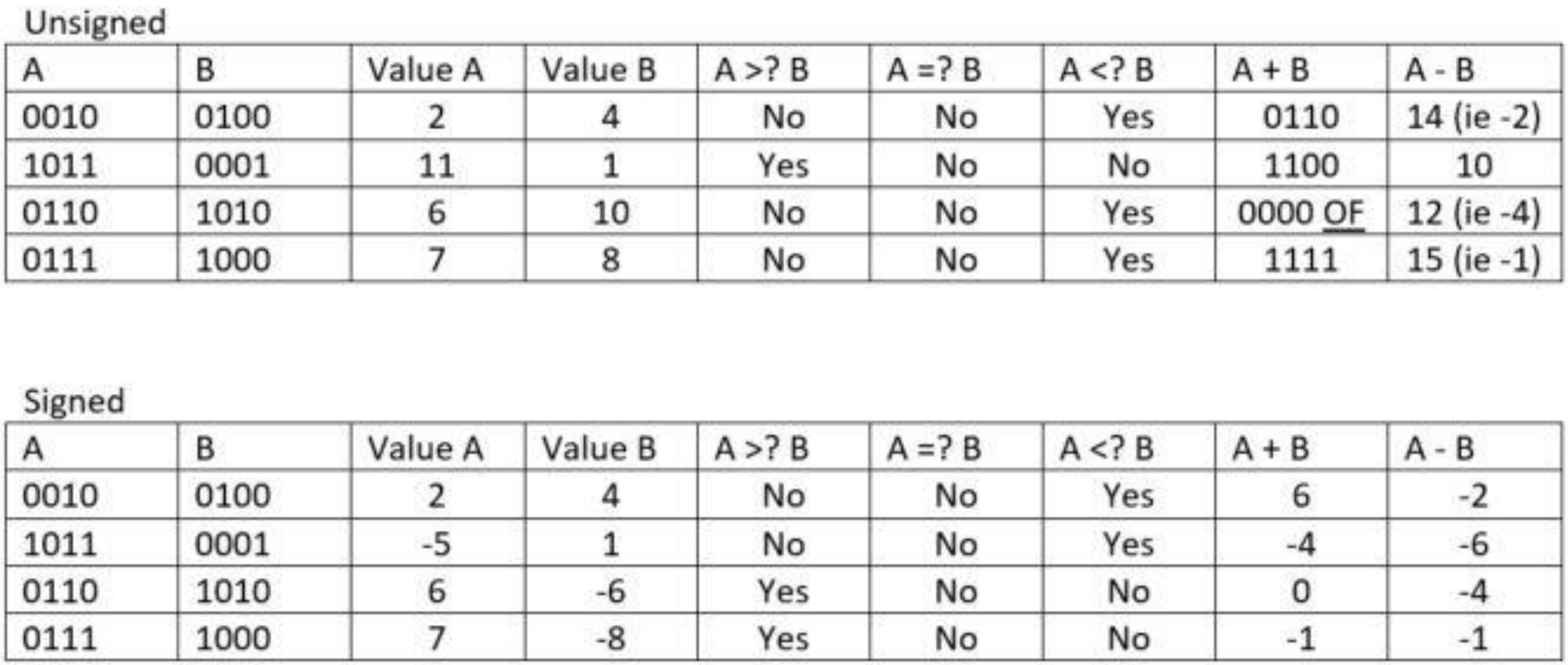
clue below: brute force?

decimal

i	17*i	hex	binary
1	17	11	10001
2	34	22	100010
3	51	33	110011
4	68	44	1000100
5	85	55	1010101
6	102	66	1100110
7	119	77	1110111
8	136	88	10001000
9	153	99	10011001
10	170	AA	10101010
11	187	BB	10111011
12	204	CC	11001100
13	221	DD	11011101
14	238	EE	11101110
15	255	FF	11111111

Teams Demo Folder?

Solution to worksheet



Unsigned

A	B	Value A	Value B	A >? B	A =? B	A <? B	A + B	A - B
0010	0100	2	4	No	No	Yes	0110	14 (ie -2)
1011	0001	11	1	Yes	No	No	1100	10
0110	1010	6	10	No	No	Yes	0000 OF	12 (ie -4)
0111	1000	7	8	No	No	Yes	1111	15 (ie -1)

Signed

A	B	Value A	Value B	A >? B	A =? B	A <? B	A + B	A - B
0010	0100	2	4	No	No	Yes	6	-2
1011	0001	-5	1	No	No	Yes	-4	-6
0110	1010	6	-6	Yes	No	No	0	-4
0111	1000	7	-8	Yes	No	No	-1	-1