CSU22022 Computer Architecture I

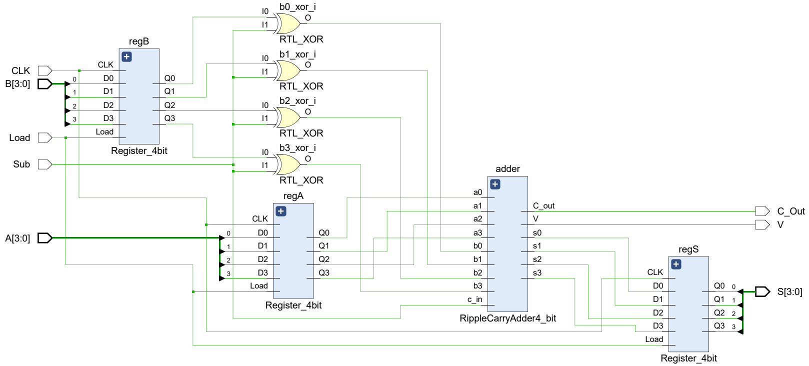
Twelfth Lecture - Adder-Subtractor

Michael Manzke

2023-2024

Trinity College Dublin

Adder-Subtractor (4-bit)



How to subtract

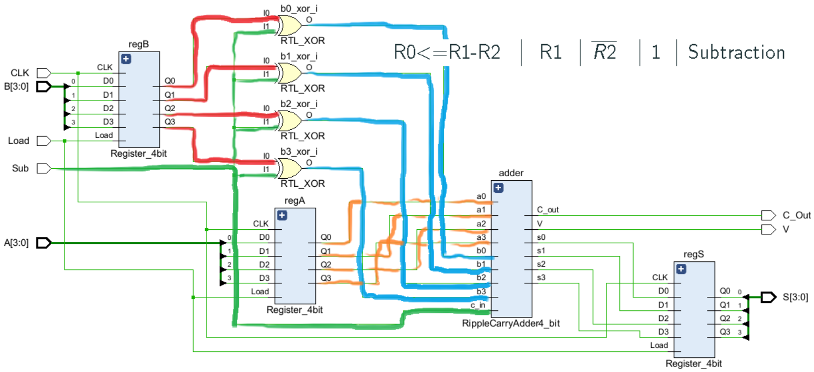


Figure 1: We use XOR-gates for every bit of the B-vector. If the signal "S"='0' a '1' remains a '1' and a '0' remains a '0', but if the signal "S"='1', a '1' becomes a '0' and a '0' becomes a '1'. Therefore, computing the 1's complement. Furthermore, we use the signal "S" to set the carry input signal "C" of the ripple carry adder. Therefore, adding a '1'. Consequently, we have the 1's Complement plus '1'. This makes 2's complement numbers negative if they were positive and positive numbers negative.

2's Complement Range of a n-bit Register $2^{n-1} - 1$ to -2^{n-1}

Two's Complement	Decimal		
011_2	3_{10}	$2^{n-1} - 1$	$2^{3-1} - 1 = 3$
010_2	2_{10}		
001_2	1_{10}		
000_2	0_{10}		
111_2	-1_{10}		
110_2	-2_{10}		
101_2	-3_{10}		
100_2	-4_{10}	-2^{n-1}	$-2^{3-1} = -4$

Table 1: For a fixed size register for arithmetic operands there is the hazard of overflow.

Overflow and Carry $K_1 : C \leftarrow C_n, V \leftarrow C_n \oplus C_{n-1}$

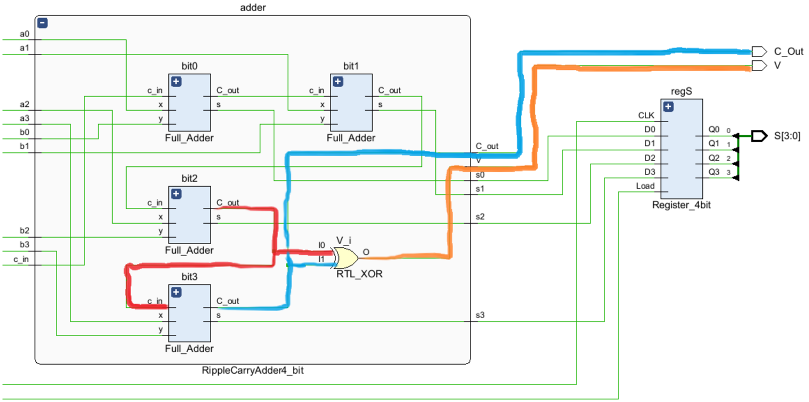


Figure 2: We can detect a overflow or a carry condition by recording the Status bits C=Carry and V=Overflow.

Status Register - V, C, N, and Z Flags

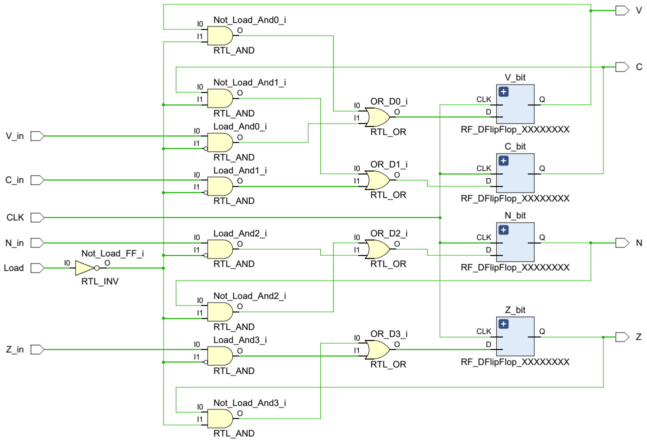


Figure 3: We should implement a status register to recording the Status bits C=Carry, V=Overflow, N=Negative, and Z=Zero. The Status flags allow to implement control flow.

Status Register - One

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity Register_Status is
    Port ( V_in, C_in, N_in, Z_in : in STD_LOGIC;
           CLK, Load : in STD_LOGIC;
           V, C, N, Z : out STD_LOGIC);
end Register_Status;

architecture Behavioral of Register_Status is

    COMPONENT RF_DFlipFlop_XXXXXXXX
    Port ( CLK, D : in STD_LOGIC;
           Q : out STD_LOGIC);
    END COMPONENT;
```

Listing 1: entity Register_Status

Status Register - Two

```vhdl
    signal Q_bit0, Q_bit1, Q_bit2, Q_bit3 : std_logic;
    signal OR_D0, OR_D1, OR_D2, OR_D3 : std_logic;
    signal Not_Load_FF, Load_FF : std_logic;
    signal Not_Load_And0, Not_Load_And1, Not_Load_And2,
    Not_Load_And3 : std_logic;
    signal Load_And0, Load_And1, Load_And2, Load_And3 :
    std_logic;

    --  Propagation Delay according to StdentID e.g. 26 33
    57 25(DEC)
    constant AND_gate_delay : Time := 6ns; -- 6 =5+1
    constant NAND_gate_delay : Time := 3ns;-- 3=2+1
    constant OR_gate_delay : Time := 8ns;  -- 8=7+1
    constant NOR_gate_delay : Time := 6ns; -- 6=5+1
    constant XOR_gate_delay : Time := 4ns; -- 4=3+1
    constant XNOR_gate_delay : Time := 4ns;-- 4=3+1
    constant NOT_gate_delay : Time := 7ns; -- 7=6+1
```

Listing 2: signals and constants

Status Register - Three

```vhdl
begin

   Not_Load_FF <= not Load after NOT_gate_delay;
   Load_FF <= not Not_Load_FF after NOT_gate_delay;

   -- Instantiate the Overflow Status Bit V
   V_bit: RF_DFlipFlop_XXXXXXXX PORT MAP (
          CLK => CLK, D => OR_D0, Q => Q_bit0 );

   Not_Load_And0 <= Q_bit0 and Not_Load_FF after
    AND_gate_delay;
   OR_D0 <= Not_Load_And0 or Load_And0 after OR_gate_delay;
   Load_And0 <= V_in and Load_FF after AND_gate_delay;
   V <= Q_bit0;
```

Listing 3: Overflow Status Bit V

Status Register - Four

```
    -- Instantiate the Carry Status Bit C
    C_bit: RF_DFlipFlop_XXXXXXXX PORT MAP (
           CLK => CLK, D => OR_D1, Q => Q_bit1 );

    Not_Load_And1 <= Q_bit1 and Not_Load_FF after
     AND_gate_delay;
    OR_D1 <= Not_Load_And1 or Load_And1 after OR_gate_delay;
    Load_And1 <= C_in and Load_FF after AND_gate_delay;
    C <= Q_bit1;

    -- Instantiate the Negative Status Bit N
    N_bit: RF_DFlipFlop_XXXXXXXX PORT MAP (
           CLK => CLK, D => OR_D2, Q => Q_bit2);

    Not_Load_And2 <= Q_bit2 and Not_Load_FF after
     AND_gate_delay;
    OR_D2 <= Not_Load_And2 or Load_And2 after OR_gate_delay;
    Load_And2 <= N_in and Load_FF after AND_gate_delay;
    N <= Q_bit2;
```

Listing 4: Status Bits C and N

Status Register - Five

```vhdl
    -- Instantiate the Zero Status Bit Z
    Z_bit: RF_DFlipFlop_XXXXXXXX PORT MAP (
           CLK => CLK, D => OR_D3, Q => Q_bit3);

    Not_Load_And3 <= Q_bit3 and Not_Load_FF after
     AND_gate_delay;
    OR_D3 <= Not_Load_And3 or Load_And3 after OR_gate_delay;
    Load_And3 <= Z_in and Load_FF after AND_gate_delay;
    Z <= Q_bit3;

end Behavioral;
```

Listing 5: Zero Status Bit Z

Adder-Subtractor with Status Register

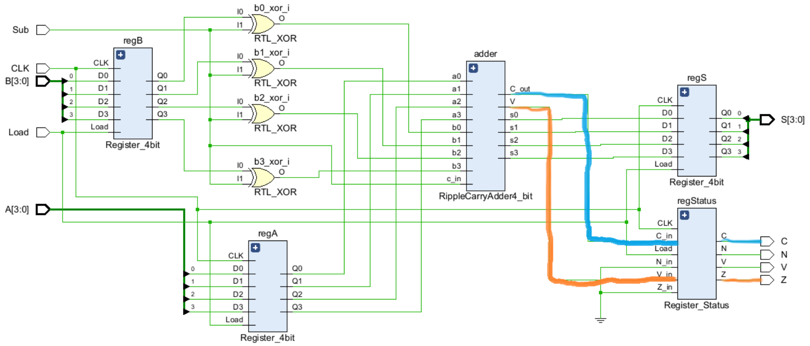


Figure 4: The status register is recording the Status bits C=Carry and V=Overflow. Also, we need to implement the N=Negative and Z=Zero.

Adder-Subtractor

This then is the basis for the adder-subtractor. The control input $\overline{X}$ selects addition and X subtraction. In our example X is the "Sub" signal and K_1 the Load signal.

$$\overline{X}.K_1 : R2 \leftarrow R0 + R1 \tag{1}$$

$$X.K_1 : R2 \leftarrow R0 + \overline{R1} + 1 \tag{2}$$

$$K_1 : C \leftarrow C_n, V \leftarrow C_n \oplus C_{n-1} \tag{3}$$

2's Complement N status Bit

Two's Complement	Decimal		
011_2	3_{10}	$2^{n-1} - 1$	$2^{3-1} - 1 = 3$
010_2	2_{10}		
001_2	1_{10}		
000_2	0_{10}		
111_2	-1_{10}		
110_2	-2_{10}		
101_2	-3_{10}		
100_2	-4_{10}	-2^{n-1}	$-2^{3-1} = -4$

Table 2: If the most significant bit (MSB) is a one we have a negative number.

The N Status bit

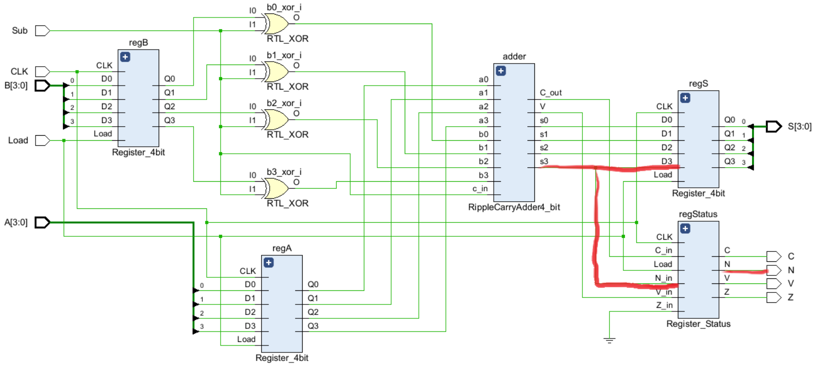


Figure 5: We connect the the MSB to the N-bit in the status register.

The Z Status bit

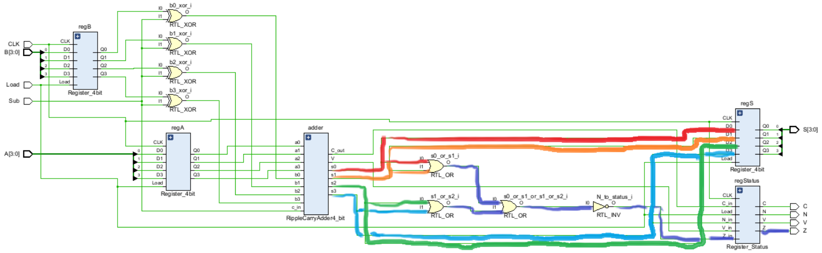


Figure 6: We "or" together all sum-bits, "not" the output, and connect it to the Z-bit in the status register. Therefore, if all bits are zero the output is one.

Add and Subtract Micro-operation

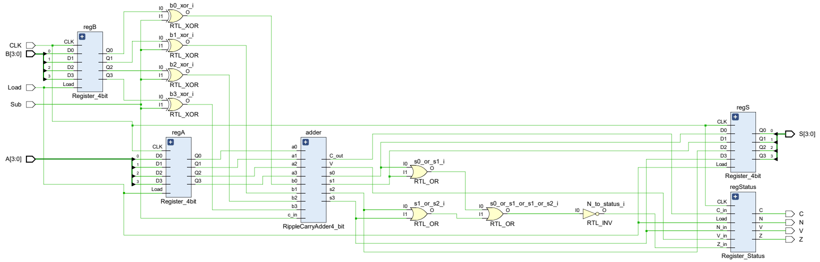


Figure 7: This synchronous circuit adds data from register B to data in register A or subtracts data from register B from data in register A and writes the result into register S. Furthermore, it calculates the status-flags V, C, N, and Z and writes these into the status register.

Add-Subtract Micro-operation executed in 1 Clock Cycle

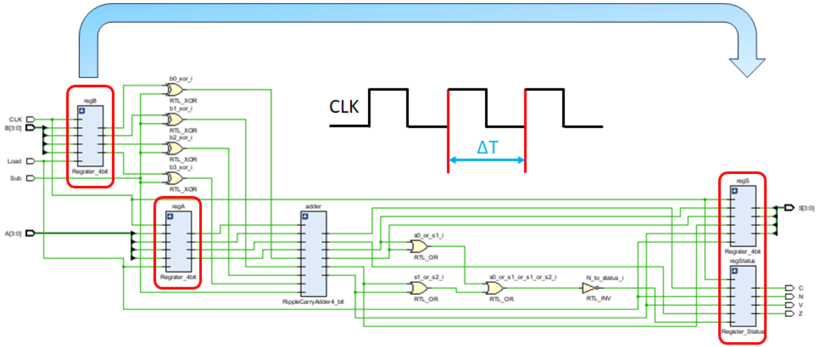


Figure 8: This circuit propagates data from registers A and B through the Ripple-Carry-Adder and its associated logic into register S and the status register in 1 clock cycle. The Ripple-Carry-Adder's and associated logic's worst case propagation delay plus the propagation delay of the receiving registers must be less than the ΔT of the clock.