Base version of the nano processor

Instructions saved in the program ROM

```
"101110000001",     -- MOV R7 1
"101100000010",     -- MOV R6 2
"101010000011",     -- MOV R5 3
"001111100000",     -- ADD R7 R6
"001111010000",     -- ADD R7 R5
"110100000101"      -- JZR R2 101
```

Instruction	Description	Format (12-bit instruction)
MOV R, d	Move immediate value *d* to register R.	1 0 R R R 0 0 0 d d d d
ADD Ra, Rb	Add values in registers Ra and Rb and store the result in Ra.	0 0 Ra Ra Ra Rb Rb Rb 0000
NEG R	2's complement of registers R	0 1 R R R 0 0 0 0 0 0 0
JZR R, d	Jump if value in register R is 0, i.e., If R == 0 PC <= d; Else PC <= PC + 1;	1 1 R R R 0 0 0 0 d d d

Expected Behaviour

When the bit stream is first loaded and when the reset button is pressed (reset button must be hold for 1/2 seconds) the above program will run from the beginning. Since the leds and the seven segment will show the values in the R7 register first it will show zero, then one, then three and lastly six. No flags will be shown. Leds will show the values in binary format and the seven segment display will show the values in decimal.
If the program doesn't run as soon as the bit stream is loaded try pushing the Reset (U18) button for 1/2 second.

Port Mapping

LED
LED[0] <= First bit in the R7 register value
LED[1] <= Second bit in the R7 register value
LED[2] <= Third bit in the R7 register value
LED[3] <= Fourth bit in the R7 register value

LED[14] <= Zero flag
LED[15] <= Overflow flag

Buttons
U18 <= Reset button

The value in the output is shown in the first of the four seven segment displays. You can see the relevant constraints file below.

```
##7 segment display
set_property PACKAGE_PIN W7 [get_ports {R7_Display[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {R7_Display[0]}]
set_property PACKAGE_PIN W6 [get_ports {R7_Display[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {R7_Display[1]}]
set_property PACKAGE_PIN U8 [get_ports {R7_Display[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {R7_Display[2]}]
set_property PACKAGE_PIN V8 [get_ports {R7_Display[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {R7_Display[3]}]
set_property PACKAGE_PIN U5 [get_ports {R7_Display[4]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {R7_Display[4]}]
set_property PACKAGE_PIN V5 [get_ports {R7_Display[5]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {R7_Display[5]}]
set_property PACKAGE_PIN U7 [get_ports {R7_Display[6]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {R7_Display[6]}]

#set_property PACKAGE_PIN V7 [get_ports dp]
    #set_property IOSTANDARD LVCMOS33 [get_ports dp]

set_property PACKAGE_PIN U2 [get_ports {Anode[0]}]              #Anode <= "1110"
    set_property IOSTANDARD LVCMOS33 [get_ports {Anode[0]}]
set_property PACKAGE_PIN U4 [get_ports {Anode[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Anode[1]}]
set_property PACKAGE_PIN V4 [get_ports {Anode[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Anode[2]}]
set_property PACKAGE_PIN W4 [get_ports {Anode[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Anode[3]}]
```

Modified version of the nano processor

Instructions saved in the program ROM

```
"00100100000010",       -- MOV R2 2
"00101110000001",       -- MOV R7 1
"01101110100000",       -- MUL R7 R2
"01101110100000",       -- MUL R7 R2
"00001110100000",       -- ADD R7 R2
"01011110010000",       -- INC R7
"01001110100000",       -- SUB R7 R2
"01001110100000",       -- SUB R7 R2
"00101100000001",       -- MOV R6 1
"11001111100000",       -- AND R7 R6
"10000101110000",       -- COMP R2 R7
"00110110001011"        -- JZR R3 11
```

Supported Instructions

Instruction	Description	Format (14-bit instruction)
MOV R, d	Move immediate value *d* to register R.	0 0 1 0 R R R 000 d d d d
ADD Ra, Rb	Add the values in registers Ra and Rb and store the result in Ra.	0000 Ra Ra Ra Rb Rb Rb 0000
NEG R	2's complement of registers R	0 0 0 1 R R R 0 0 0 0 0 0 0
JZR R, d	Jump if value in register R is 0, i.e., If R == 0 PC <= d; Else PC <= PC + 1;	0 0 1 1 R R R 0 0 0 0 d d d
SUB Ra, Rb	Subtract the values in registers Ra and Rb and store the result in Ra.	0100 Ra Ra Ra Rb Rb Rb 0000
INC Ra	Increment the value in the register Ra by 1.	0101 Ra Ra Ra 001 0000
DNC Ra	Decrement the value in the register Ra by 1.	0111 Ra Ra Ra 001 0000
MUL Ra, Rb	Multiply the values in registers Ra and Rb and store the result in Ra.	0110 Ra Ra Ra Rb Rb Rb 0000
AND Ra, Rb	Get the bit wise AND value of the values in registers Ra and Rb and store the result in the register Ra.	1100 Ra Ra Ra Rb Rb Rb 0000
OR Ra, Rb	Get the bit wise OR value of the	1101 Ra Ra Ra Rb Rb Rb

	values in registers Ra and Rb and store the result in the register Ra.	0000
NOT Ra	Get the bit wise NOT value of the value in register Ra and store the result in the register Ra.	1110 Ra Ra Ra 000 0000
XOR Ra, Rb	Get the bit wise XOR value of the values in registers Ra and Rb and store the result in the register Ra.	1111 Ra Ra Ra Rb Rb Rb 0000
COMP Ra, Rb	Compare the value in register Rb with respect to Ra and output whether its greater than, lesser than or equal Ra.	1000 Ra Ra Ra Rb Rb Rb 0000

Expected Behaviour

If the program doesn't run immediately as the bit stream is loaded try pressing reset button for 0.5 seconds. When you want to reset the processor hold the reset button for about 0.5 seconds.

Instruction in Machine Code	Instruction	Output
00100100000010	MOV R2 2	Move value 2 to the register R2. But still the output of the leds and the display is 0.
00101110000001	MOV R7 1	Move value 1 to the register R7. The value shown by leds and the display is 1.
01101110100000	MUL R7 R2	Multiply R7 by R2 and store the value in R7. So the output is 2
01101110100000	MUL R7 R2	Multiply R7 by R2 and store the value in R7. So the output is 4
00001110100000	ADD R7 R2	Add R7 and R2 and store the value in R7. So the output is 6.
01011110010000	INC R7	Increment the value in R7 by one and store it in R7. Output is 7.
01001110100000	SUB R7 R2	Subtract R2 from R7 and save the value in R7. The output is 5
01001110100000	SUB R7 R2	Subtract R2 from R7 and save the value in R7. The output is 5
00101100000001	MOV R6 1	Move the value one to register R6. No change in the output.
11001111100000	AND R7 R6	Take the bitwise AND of R7 and R6. Then store the value in R7. So the output is 1.

10000101110000	COMP R2 R7	Compare the value R7 with respect to R2. Then greater than flag will turn on which is the led 9
00110110001011	JZR R3 11	

Port Mapping

LED
LED[0] <= First bit in the R7 register value
LED[1] <= Second bit in the R7 register value
LED[2] <= Third bit in the R7 register value
LED[3] <= Fourth bit in the R7 register value

LED[9] <= Greater than flag (Output of the comparator)
LED[10] <= Equal flag (Output of the comparator)
LED[11] <= Lesser than flag (Output of the comparator)

LED[14] <= Zero flag
LED[15] <= Overflow flag

Buttons
U18 <= Reset button

The value in the output is shown in the first of the four seven segment displays. You can see the relevant constraints file below.

```
##7 segment display
set_property PACKAGE_PIN W7 [get_ports {R7_Display[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {R7_Display[0]}]
set_property PACKAGE_PIN W6 [get_ports {R7_Display[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {R7_Display[1]}]
set_property PACKAGE_PIN U8 [get_ports {R7_Display[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {R7_Display[2]}]
set_property PACKAGE_PIN V8 [get_ports {R7_Display[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {R7_Display[3]}]
set_property PACKAGE_PIN U5 [get_ports {R7_Display[4]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {R7_Display[4]}]
set_property PACKAGE_PIN V5 [get_ports {R7_Display[5]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {R7_Display[5]}]
set_property PACKAGE_PIN U7 [get_ports {R7_Display[6]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {R7_Display[6]}]

#set_property PACKAGE_PIN V7 [get_ports dp]
    #set_property IOSTANDARD LVCMOS33 [get_ports dp]

set_property PACKAGE_PIN U2 [get_ports {Anode[0]}]              #Anode <= "1110"
    set_property IOSTANDARD LVCMOS33 [get_ports {Anode[0]}]
set_property PACKAGE_PIN U4 [get_ports {Anode[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Anode[1]}]
set_property PACKAGE_PIN V4 [get_ports {Anode[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Anode[2]}]
set_property PACKAGE_PIN W4 [get_ports {Anode[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Anode[3]}]
```