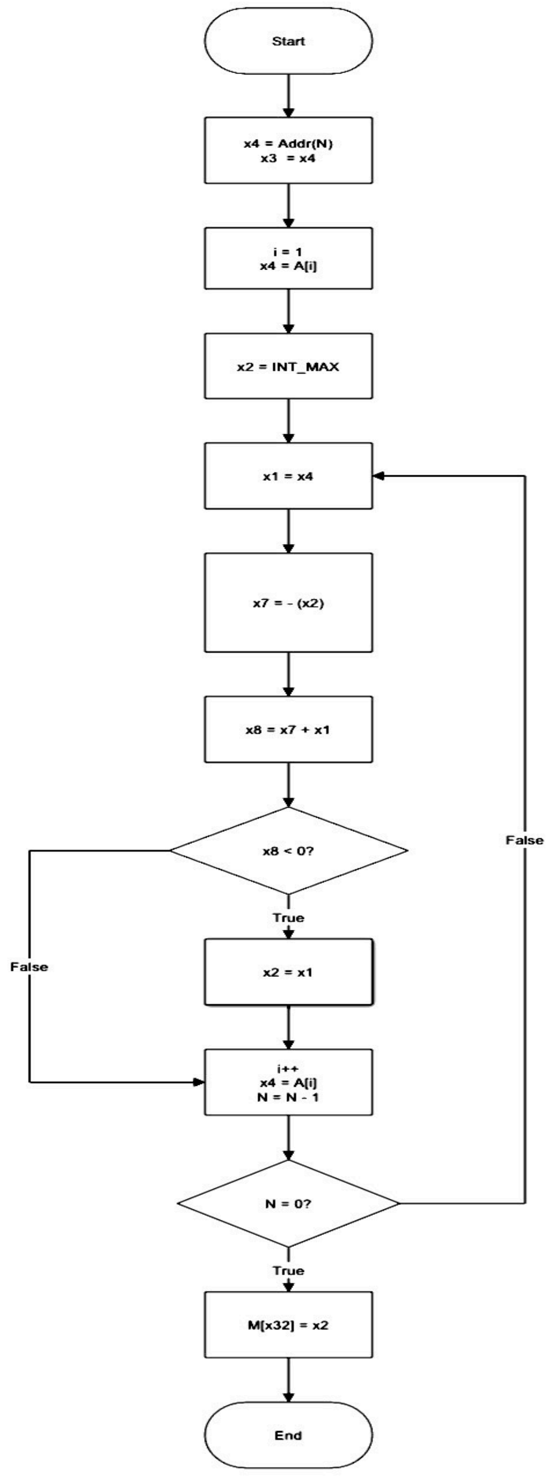
Assumptions

1. The input array has at least one element i.e, the input array A is non-empty
2. The address of size of input array is stored in register x4
3. The input array is stored in memory location following x4. Memory is named M
4. When the execution of the program begins, the program counter (PC) points to the first instruction
5. Register x0 is reserved for the constant 0

Flow Chart



Assembly Code

Benchmark for Version 1: To find minimum number out of ‘n’ numbers without using label(s)

x1 - Current input value
x2 - Minimum value
x3 - Size of the input array, counter
x4 - Address of the size of the array
x6 - Start of the loop
x8 - Sum of current input value and minimum value
x10 - Position of array increment and counter decrement
x11 - Position of update minimum value
x12 - Position of storing the final minimum number
M[x32] - Memory location where final minimum value is stored

```
1. SVPC x40, x0              // save the address of the first line of program in register x40
2. INC x10, x40, 0x39.       // calculate offset of line 13
3. INC x11, x40, 0x49        // calculate offset of line 17
4. INC x12, x40, 0x59        // calculate offset of line 19
5. LD x3, x4                 // load the size of input array to x3 from x4
6. INC x4, x4, 0x4           // move current position to the first element of the array
7. INC x2, x0, 0x7FFF FFFF   // x2 is initialized with INT_MAX
8. SVPC x6, 0x4              // go to the start of the loop in memory stack
9. LD x1, x4                 // store current input value in x1
10. NEG x7, x2               // negate the value in x2 and store in x7
11. ADD x8, x7, x1           // x8 = value in x7 + current input in x1
12. BRN x11                  // if min + current input < 0, go to update min
13. INC x4, x4, 0x4          // increment array position to point to next element
14. INC x3, x3, 0xFFFF FFFF  // decrement the counter
15. BRZ x12                  // if no more array elements are present, jump to the last line
16. J x6                     // else go to the start of the loop
17. INC x2, x1, 0x0          // update the minimum value in x2
18. J x10                    // jump to increment array position and decrement the counter
19. ST x2, x32               // store the final minimum value in M[x32]
```

Benchmark for Version 2: To find minimum number out of ‘n’ numbers using MIN instruction

x1 - Minimum value
x3 - Size of the input array, counter
x4 - Address of the size of the array
M[x32] - Memory location where final minimum value is stored

```
1. LD x3, x4          // load the size of array from x4 to x3
2. INC x4, x4, 0x4    // move current position to the first element of the array
3. MIN x1, x4, x3     // find minimum element of the array using MIN
4. ST x1, x32         // store the minimum value in M[x32]
```