

# EMBEDDED SYSTEM PROGRAMMING - 1.1

August 20, 2021



Sindri Thor Jonsson

# 1 PART

**TASK:** Measure the memory footprint of an array of 1000 integers of default size (int) as well as size 8, 16, 32, 64, using the size information printed at the end of the compilation. In each case, indicate whether or not the available processor memory is exceeded. From the above measurements, infer how many bytes the compiler allocates for a single int variable is on this processor.

**SOLUTION:** In C++ we can declare which size of integer we want to use (int8\_t, int16\_t, int32\_t, int64\_t). When using platform IO the memory usage is written to the terminal after building.

The available processor memory (RAM) on the Arduino Nano is 2048 bytes.

**These are the results of creating an array of 1000 integers of the sizes (8,16,32,64bits):**

```
Linking .pio\build\nanoatmega328\firmware.elf
Checking size .pio\build\nanoatmega328\firmware.elf
Advanced Memory Usage is available via "PlatformIO Home > Project Inspect"
RAM: [=====] 97.7% (used 2000 bytes from 2048 bytes)
Flash: [ ] 0.7% (used 216 bytes from 30720 bytes)
Building .pio\build\nanoatmega328\firmware.hex
===== [SUCCESS] Took 1.24 seconds =====
```

(a) Default size int

```
Linking .pio\build\nanoatmega328\firmware.elf
Checking size .pio\build\nanoatmega328\firmware.elf
Advanced Memory Usage is available via "PlatformIO Home > Project Inspect"
RAM: [=====] 48.8% (used 1000 bytes from 2048 bytes)
Flash: [ ] 0.7% (used 210 bytes from 30720 bytes)
Building .pio\build\nanoatmega328\firmware.hex
===== [SUCCESS] Took 1.29 seconds =====
```

(b) 8-bit int

```
Linking .pio\build\nanoatmega328\firmware.elf
Checking size .pio\build\nanoatmega328\firmware.elf
Advanced Memory Usage is available via "PlatformIO Home > Project Inspect"
RAM: [=====] 97.7% (used 2000 bytes from 2048 bytes)
Flash: [ ] 0.7% (used 216 bytes from 30720 bytes)
Building .pio\build\nanoatmega328\firmware.hex
===== [SUCCESS] Took 1.24 seconds =====
```

(a) 16-bit int

```
Linking .pio\build\nanoatmega328\firmware.elf
Checking size .pio\build\nanoatmega328\firmware.elf
Advanced Memory Usage is available via "PlatformIO Home > Project Inspect"
RAM: [=====] 195.3% (used 4000 bytes from 2048 bytes)
Flash: [ ] 0.7% (used 224 bytes from 30720 bytes)
Building .pio\build\nanoatmega328\firmware.hex
===== [SUCCESS] Took 1.25 seconds =====
```

(b) 32-bit int

```
Linking .pio\build\nanoatmega328\firmware.elf
Checking size .pio\build\nanoatmega328\firmware.elf
Advanced Memory Usage is available via "PlatformIO Home > Project Inspect"
RAM: [=====] 390.6% (used 8000 bytes from 2048 bytes)
Flash: [ ] 0.8% (used 250 bytes from 30720 bytes)
Building .pio\build\nanoatmega328\firmware.hex
===== [SUCCESS] Took 1.33 seconds =====
```

Figure 3: 64-bit int

From these results we can see that the memory footprint from the int types, the compiler allocates 1 byte for each the 8 bit int. 2 bytes for the default int size which is the 16bit int and 4 bytes for each int of 64b it size.

When creating the array of 1000 integers in 32bit or 64bit size the available processor memory is exceeded.

## 2 PART

**GLOBAL:** Using standard integer array of 1000 integers and global declaration with a return statement of the array in the main function the RAM allocation is 2000bytes + the flash memory usage increases by 2000bytes. Using constant double did not add anything to the flash nor RAM usage

**LOCAL:** Using standard integer array of 1000 integers and local declaration with a return statement of the array in the main function the RAM and flash memory allocated does not change. Static int array declared locally added 2000bytes of RAM usage. Dynamically allocated integer array of 1000 integers allocated only 10bytes of RAM but around 800bytes of flash memory. Im pretty sure these results are not correct or making sense but i did not find a fix unfortunately.