# LAB REPORT

**Lab Objectives:**

• Learn how to configure the 8051 for serial communication and how to write serial device drivers.

• Learn how to use internal XRAM and external XRAM (using the NVSRAM).

• Begin learning how to use a compiler (e.g. SDCC) to develop C programs.

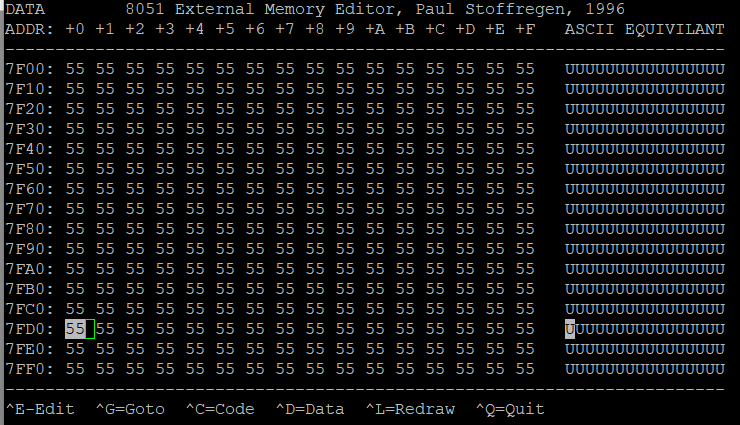
• Learn how to initialize hardware properly in C.

• Continue learning about the ARM architecture and development environment

## PART 1

Enabled the XSR1 and XSR0 bit fields in AUX register to increase the space available on XRAM to 1024 bits. Paulmon code is able to write to the entire XRAM space (0x0000 to 0xFFFF) with value 0x55.

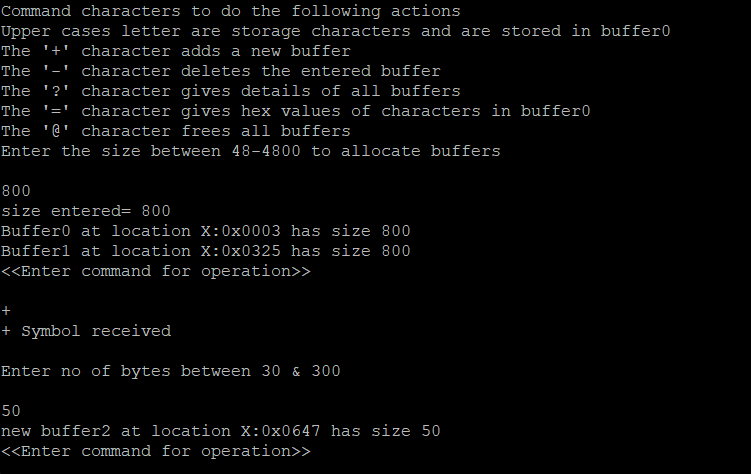
Using PAULMON2, the highest demonstrated baud rate is 57600. This was achieved through trial and error.



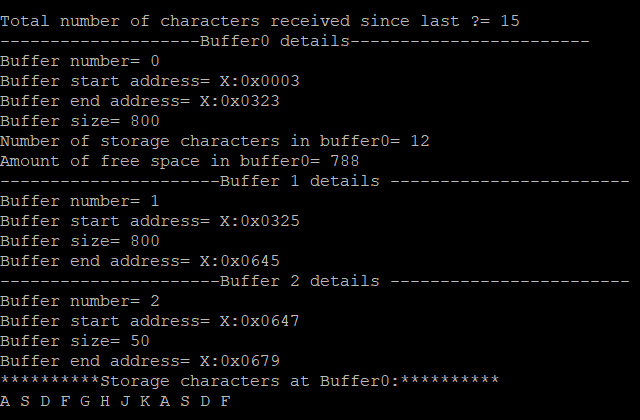
## PART 2

**SDCC linker settings** used for heap allocation: --code-loc 0x0000 --code-size 0x8000 --xram-loc 0x0001 --xram-size 0x8000 --model-large --out-fmt-ihx

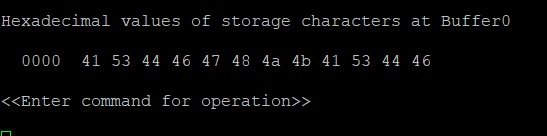
The user interface for the heap allocation is shown in the image below. Various options are presented to allocate memory in the heap, free the allocated memory, add or delete new buffers, provide heap statistics, provide a hexdump of the buffer and delete all the allocated buffers.

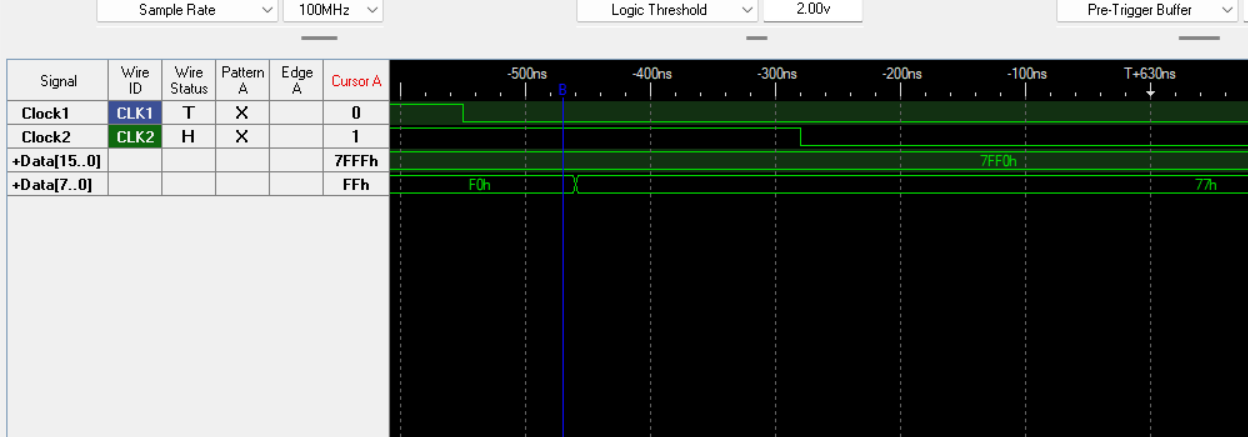


When ‘?’ character is pressed, heap statistics are printed as shown.



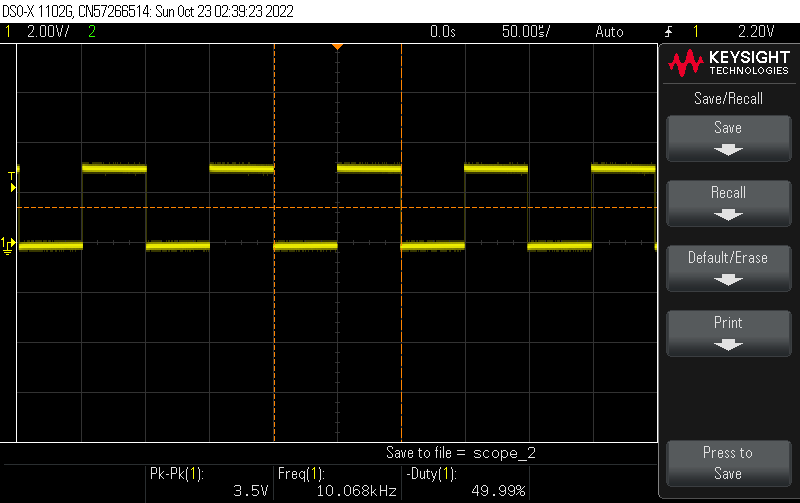
When ‘=’ character is pressed, hexadecimal values are printed as follows.



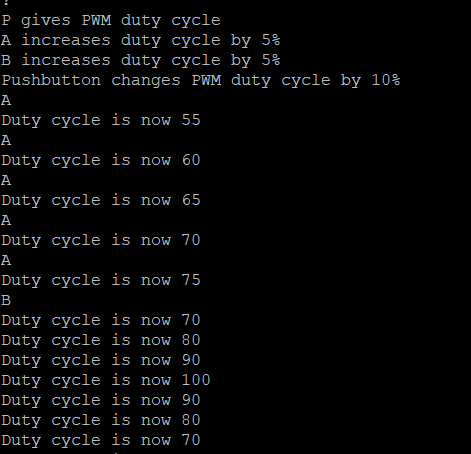
**Virtual Debug Port:** The virtual debug port i=makes debugging port easier than when using serial port. When the program is first entered the DEBUGPORT(x) function prints the value 77h to the data address 0x7FF0h. The analysis follows external memory data write cycle. When both ALE (CLK1) and /WR (CLK2) are low, the data is written into the address.

## PART 3

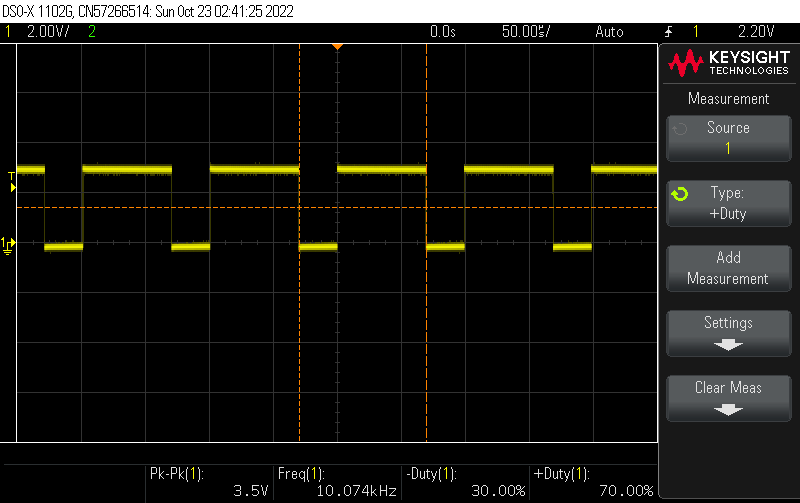
PWM signal is generated using UART, PWM module and pushbutton interrupts. The default PWM signal when the program starts is **50%:**



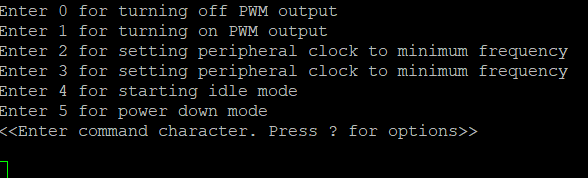
The user options to change PWM are shown when the character ‘?’ is received.



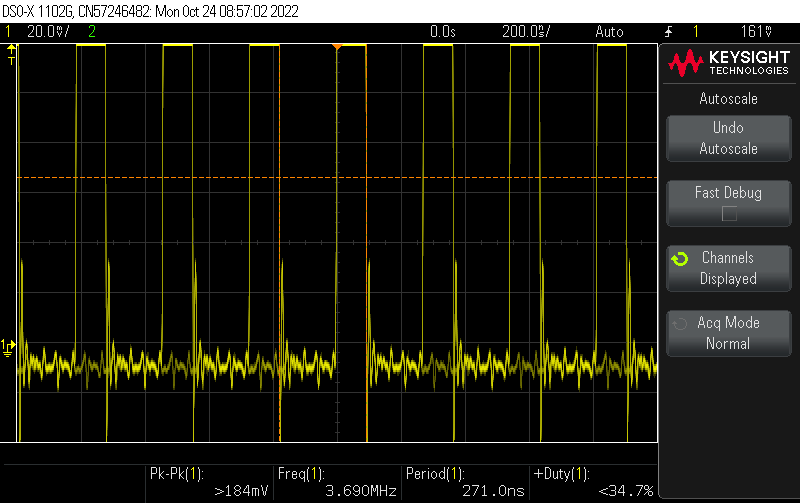
The output duty cycle after multiple user inputs through serial port (characters- ‘A’,’B’) and pushbutton is **70%.**



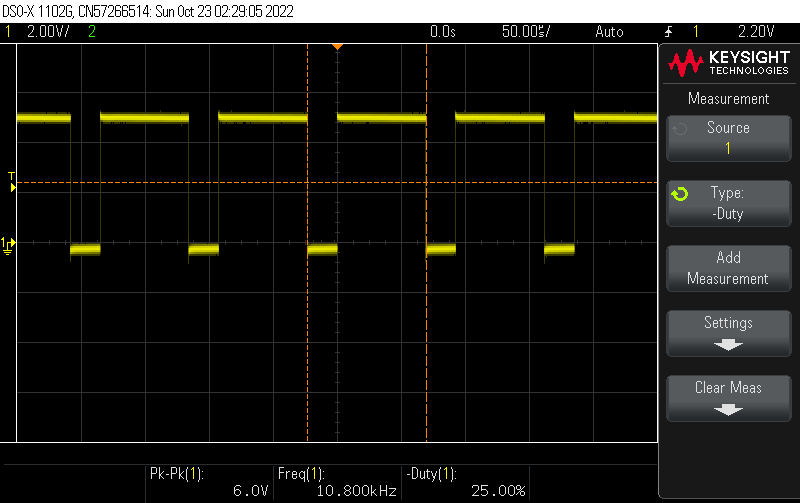
**Supplemental Element 1:** The PWM output is routed to green LED (PD12) by initializing timer 4 and its channel 1 in PWM mode.

**Supplemental Element 2:**  The user interface for PCA module shows the user options to turn on/off PWM output, set the peripheral clock to minimum and maximum frequency, setting the processor in idle mode and power down mode.

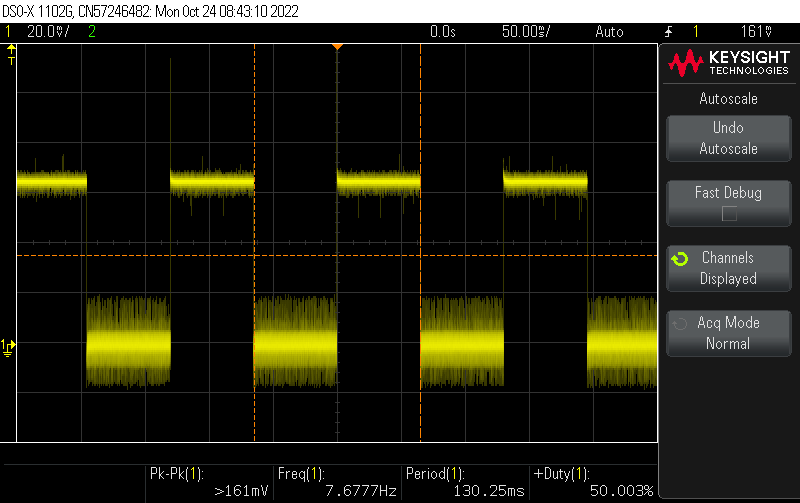
The system clock is initialised in X2 mode i.e., clock frequency is 3**.690MHz.**



Generation of PWM signal using PCA module0 in **25%** duty cycle:



Software timer is implemented using PCA module1 in interrupt mode (setting the CCF1 bit in CCON register). The port pin P1.1 connected to green LED on the board is toggled in the interrupt vector.



* Maximum frequency supported by CKRL register is 3.69MHz. The baud rate for terminal emulator in this mode should be 115200.
* Minimum frequency supported by CKRL register is 7.23kHz. The baud rate for terminal emulator in this mode should be 230.
* In IDLE mode, the ALE and PSEN are high but the peripherals still get the clock so PWM signal is still running.
* In power down mode, the oscillator is stopped. This mode is only cleared when hardware reset occurs.

**SUBMISSION QUESTIONS**

a) What operating system (including revision) did you use for your 8051 code development?

Windows 11- 64-bit operating system, x64-based processor

b) What assembler(s) (including revision) did you use?

The AS31 Assembler is used as macro assembler for this lab.

c) What ARM development tools did you use?

I used the STM32CubeIDE by ST

d) Did you install and use any other software tools to complete your lab assignment?

I downloaded Teraterm in place of PuTTY.

e) Did you experience any problems or challenges with this lab assignment or any of the software tools? If so, describe the issues.

Yes, the SDCC compiler was taking precedence of global linker setting instead of my project linker settings so not enough memory space was being allocated and I experienced some issues in seeing the written data in the virtual debug port portion.

As the PCA modules have some common registers, I had issues in initializing the modules. After hard reset from power down mode, my while loop does not function.

f) If you have any suggestions.

It could be mentioned that SDCC does not compile global variables clearly and also that malloc does not work unless the xram location is 0x0001 in the linker settings.