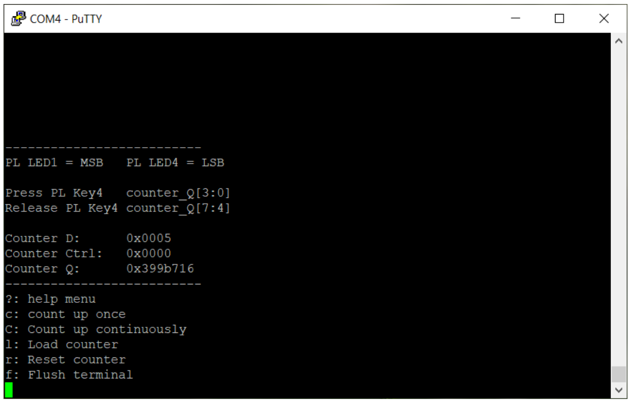
```c
#include <stdio.h>
#include "platform.h"
#include "xil_printf.h"
#include "xparameters.h"
#include "my_counter_ip.h"
#include "xuartps_hw.h"

#define COUNTER_BASEADDR      XPAR_MY_COUNTER_IP_0_S00_AXI_BASEADDR
#define USART_BASEADDR        XPAR_PS7_UART_1_BASEADDR

#define COUNTER_DREG_OFFSET   0
#define COUNTER_CTRL_OFFSET   4
#define COUNTER_QREG_OFFSET   8

#define COUNTER_HOLD_CMD      0x00           // The control bits are defined in the VHDL
#define COUNTER_LOAD_CMD      0x01           // code contained in genericCounter
#define COUNTER_COUNT_CMD     0x02           // They are added here to the bit values in
#define COUNTER_RESET_CMD     0x03           // a single place.

int main()
{
    unsigned char c;
    init_platform();
    printf("Welcome to the Counter IP\n\r");
    while(1) {
        c=XUartPs_RecvByte(USART_BASEADDR);
        switch(c) {

            case '?':
               printf("--------------------------\r\n");
               printf("PL LED1 = MSB  PL LED4 = LSB\r\n\n");
               printf("Press PL Key4         counter_Q[3:0]\r\n");
               printf("Release PL Key4       counter_Q[7:4]\r\n\n");
               printf("?: help menu\r\n");
               printf("c: count up once \r\n");
               printf("C: Count up continuously\r\n");
               printf("l: Load counter\r\n");
               printf("r: Reset counter\r\n");
               printf("f: Flush terminal\r\n");
               break;

            case 'c':
               MY_COUNTER_IP_mWriteReg(COUNTER_BASEADDR, COUNTER_CTRL_OFFSET,COUNTER_COUNT_CMD);
               MY_COUNTER_IP_mWriteReg(COUNTER_BASEADDR, COUNTER_CTRL_OFFSET,COUNTER_HOLD_CMD);
               printf("count up\r\n");
               break;

            case 'C':
               printf("Press any key to stop counting\r\n");
               MY_COUNTER_IP_mWriteReg(COUNTER_BASEADDR, COUNTER_CTRL_OFFSET,COUNTER_COUNT_CMD);
               c=XUartPs_RecvByte(USART_BASEADDR);
               MY_COUNTER_IP_mWriteReg(COUNTER_BASEADDR, COUNTER_CTRL_OFFSET,COUNTER_HOLD_CMD);
               break;

            case 'l':
               printf("Enter a 0-9 value to store in the counter: \r\n");
               c=XUartPs_RecvByte(USART_BASEADDR);
               printf("%c\r\n",c);
               MY_COUNTER_IP_mWriteReg(COUNTER_BASEADDR, COUNTER_DREG_OFFSET,c-'0');
               MY_COUNTER_IP_mWriteReg(COUNTER_BASEADDR, COUNTER_CTRL_OFFSET,COUNTER_LOAD_CMD);
               MY_COUNTER_IP_mWriteReg(COUNTER_BASEADDR, COUNTER_CTRL_OFFSET,COUNTER_HOLD_CMD);
               printf("loaded: %c\r\n",c);
               break;

            case 'r':
               MY_COUNTER_IP_mWriteReg(COUNTER_BASEADDR, COUNTER_CTRL_OFFSET,COUNTER_RESET_CMD);
               MY_COUNTER_IP_mWriteReg(COUNTER_BASEADDR, COUNTER_CTRL_OFFSET,COUNTER_HOLD_CMD);
               printf("reset\r\n");
               break;

        } // end case
    } // end while
    cleanup_platform();
    return 0;
} // end main
```

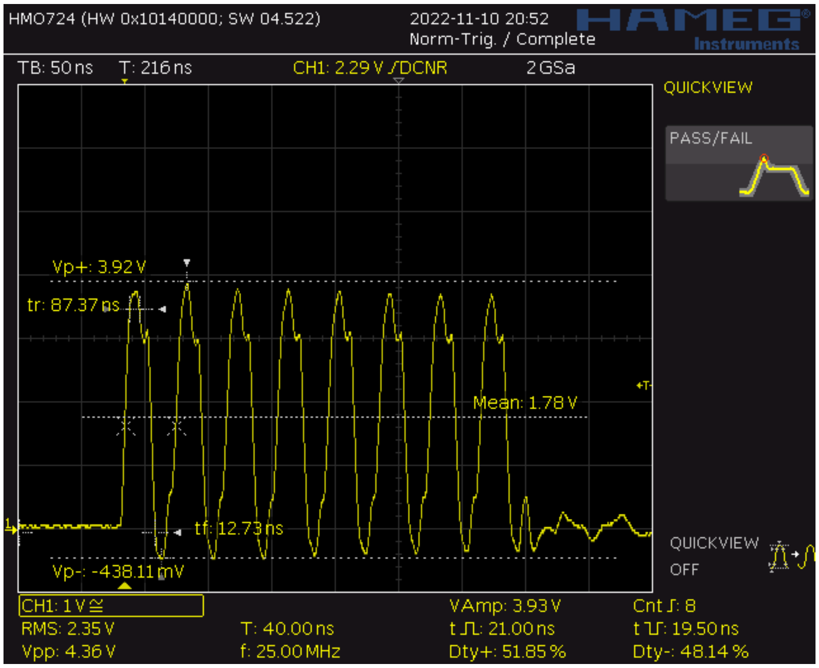
The terminal interface is a standard "?" menu interface.



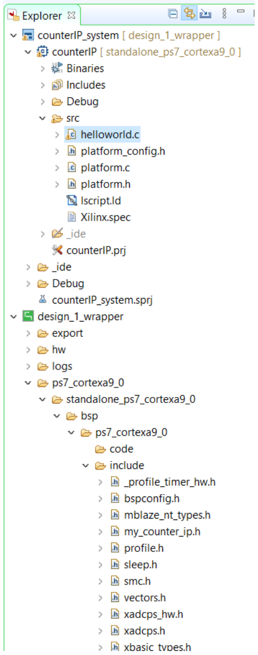
```
-------------------------
PL LED1 = MSB   PL LED4 = LSB

Press PL Key4   counter_Q[3:0]
Release PL Key4 counter_Q[7:4]

Counter D:      0x0005
Counter Ctrl:   0x0000
Counter Q:      0x399b716
-------------------------
?: help menu
c: count up once
C: Count up continuously
l: Load counter
r: Reset counter
f: Flush terminal
```

The following oscilloscope screen shot was taken while:
        Probing the anode of PL LED4 with channel 1
        Pressing PL KEY4 and then
        Pressing "c" in the PuTTY terminal

This results in the "count up" control signal to be asserted for 16 clock cycles.



```
HMO724 (HW 0x10140000; SW 04.522)   2022-11-10 20:52
                                    Norm-Trig. / Complete
TB: 50 ns   T: 216 ns        CH1: 2.29 V /DCNR       2 GSa

CH1: 1 V ≅                              VAmp: 3.93 V        Cnt ⎍: 8
RMS: 2.35 V        T: 40.00 ns          t ⎍: 21.00 ns       t ⊔: 19.50 ns
Vpp: 4.36 V        f: 25.00 MHz         Dty+: 51.85 %       Dty-: 48.14 %
```


Explorer
counterIP_system [design_1_wrapper]
counterIP [standalone_ps7_cortexa9_0]
Binaries
Includes
Debug
src
helloworld.c
platform_config.h
platform.c
platform.h
lscript.ld
Xilinx.spec
_ide
counterIP.prj
_ide
Debug
counterIP_system.sprj
design_1_wrapper
export
hw
logs
ps7_cortexa9_0
standalone_ps7_cortexa9_0
bsp
ps7_cortexa9_0
code
include
_profile_timer_hw.h
bspconfig.h
mblaze_nt_types.h
my_counter_ip.h
profile.h
sleep.h
smc.h
vectors.h
xadcps_hw.h
xadcps.h
xbasic_types.h