Embedded Systems

Week 9: Interfaces I

Fenerbahçe University

Professor & TAs

Prof: Dr. Vecdi Emre Levent

Office: 311

Email: emre.levent@fbu.edu.tr

TA: Arş. Gör. Uğur Özbalkan

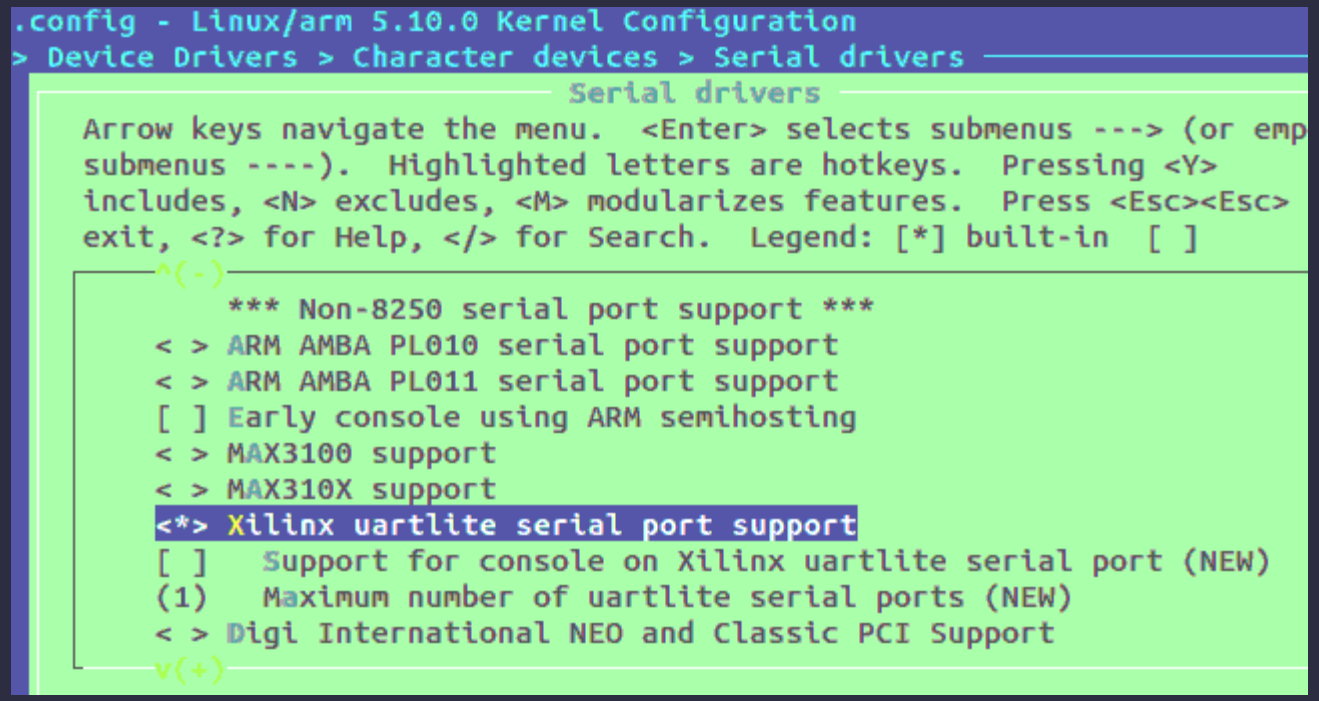
Office: 311

Email: ugur.ozbalkan@fbu.edu.tr

Petalinux

AXI UartLite

petalinux-config -c kernel



Device Drivers ---> Character devices ---> Serial drivers ---> Xilinx uartlite serial port support

Change maximum number of uartlite serial ports to 16

Petalinux

AXI UartLite

- On the PL Side, UART Interrupt must be connected.

Check kernel config page:

```
CONFIG_SERIAL_UARTLITE=y
```

- `Test from console`

echo 123456789 > /dev/ttyUL0
cat /dev/ttyUL0

Petalinux

AXI UARTLite C Example

```
#define TERMINAL    "/dev/ttyUL1"

#include <errno.h>
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <termios.h>
#include <unistd.h>

int main()
{
    char *portname = TERMINAL;
    int fd;
    int wlen;
    char *xstr = "Hello!\n";
    int xlen = strlen(xstr);

fd = open(portname, O_RDWR | O_NOCTTY
| O_SYNC);
    if (fd < 0) {
        printf("Error opening %s: %s\n",
portname, strerror(errno));
        return -1;
    }

    wlen = write(fd, xstr, xlen);
    if (wlen != xlen) {
        printf("Error from write: %d, %d\n",
wlen, errno);
    }


    /* simple noncanonical input */
 do {
        unsigned char buf[80];
#endifint rdlen;

        rdlen = read(fd, buf, sizeof(buf) - 1);
        if (rdlen > 0) {
#ifdef DISPLAY_STRING
            buf[rdlen] = 0;
            printf("Read %d: \"%s\"\n", rdlen, buf);
#else /* display hex */
            unsigned char    *p;
            printf("Read %d:", rdlen);
            for (p = buf; rdlen-- > 0; p++)
                printf(" 0x%x", *p);
            printf("\n");

        } else if (rdlen < 0) {
            printf("Error from read: %d: %s\n", rdlen,
strerror(errno));
        } else {  /* rdlen == 0 */
            printf("Timeout from read\n");
        }
        /* repeat read to get full message */
    } while (1);
}
```

Petalinux

Access MMAP function

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>
#include <sys/mman.h>

// Make the SDK console work in the debugger
#define printf(...) \
 fprintf(stdout, __VA_ARGS__); \
 fflush(stdout);

typedef long long int u64;
```

```
int main()
{
  unsigned int bram_size = 0x8000;
  off_t bram_pbase = 0x40000000; // physical base address
  u64 *bram64_vptr;
  int fd;
  printf("My test app\n");
  // Map the BRAM physical address into user space getting a virtual
address for it
  if ((fd = open("/dev/mem", O_RDWR | O_SYNC)) != -1) {
    printf("Opened\n");
    bram64_vptr = (u64 *)mmap(NULL, bram_size,
PROT_READ|PROT_WRITE, MAP_SHARED, fd, bram_pbase);
    printf("Opened %x\n", bram64_vptr);
    // Write to the memory that was mapped, use devmem from the
command line of Linux to verify it worked
    // it could be read back here also

    bram64_vptr[0] = 0xDEADBEEFFACEB00C;
    printf("Bye\n");
    close(fd);
  }
 }
```

Petalinux

UART Lite examples with ILA & VIO