Setup and Use of VC709 based AJIT single board computer

Madhav P. Desai, Gauri Patrikar
Department of Electrical Engg.
IIT Bombay, Mumbai India

December 15, 2021

The AJIT single board computer for the VC709 FPGA Card.

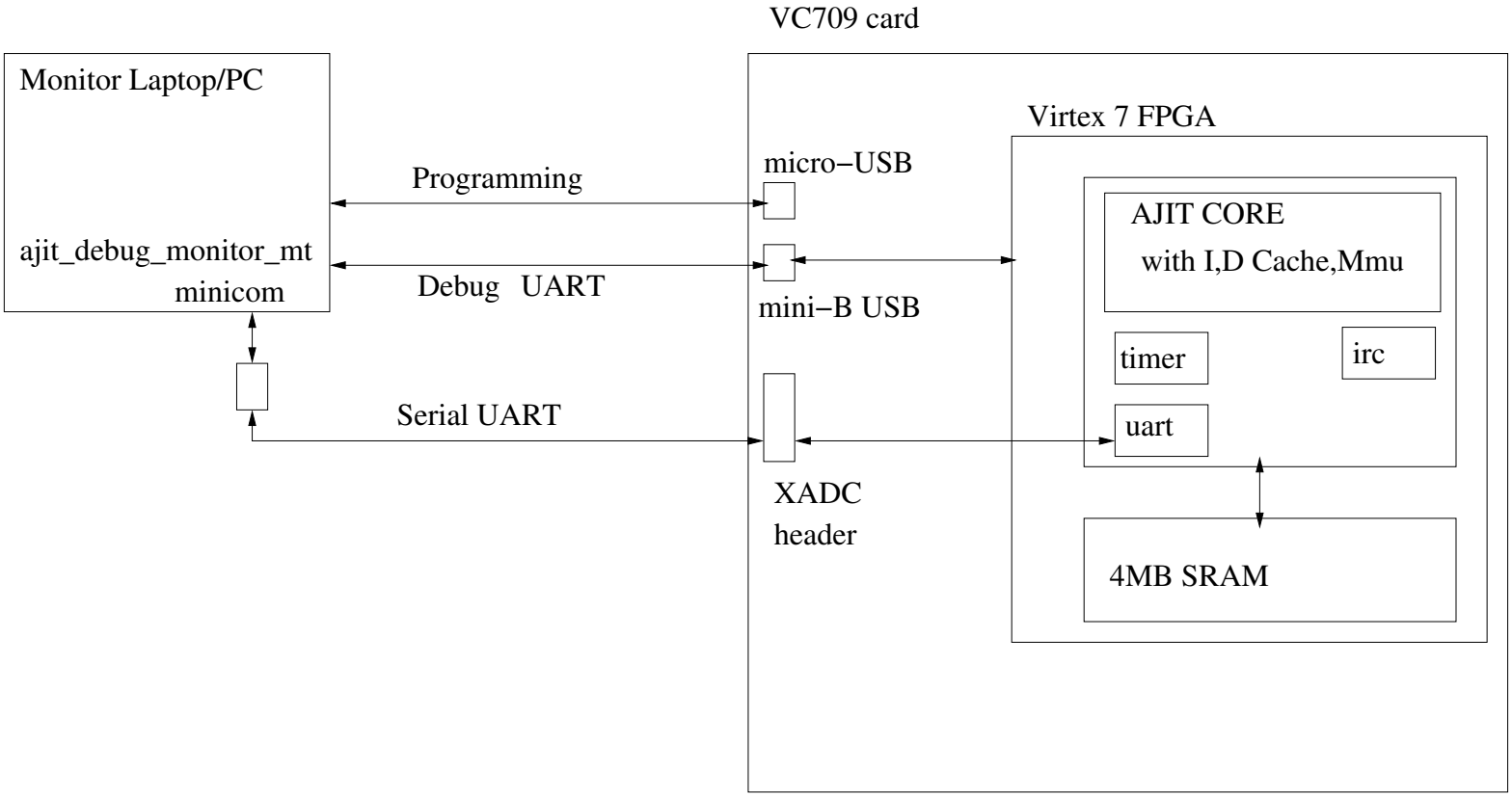

Figure: AJIT single board computer on VC709

Required Hardware

- VC709 FPGA card.
  - XADC connector must be mounted.
- Micro-USB to USB cable.
- Mini-B USB to USB cable.
- Power cable for VC709.
- UART CP210X.
- Laptop/PC with at least 3 free USB slots.
  - Running Linux.
  - We call this the monitor laptop.

Required Software

- Latest AJIT Tool Chain, installed in monitor Laptop/PC.
- Minicom terminal emulator.
- Python-3.

Making the connections

- Connect CP210X UART:
  - CP210X Ground to XADC pin 16.
  - CP210X Tx to XADC pin 18 (GPIO-O).
  - CP210X Rx to XADC pin 17 (GPIO-1).
  - Note down USB tty id. This will be used by the serial device on the VC709.
- Connect micro-USB cable between VC709 and monitor Laptop/PC.
  - Note down the new USB tty ids. These are used for programming the FPGA on the VC709.
- Connect mini-B USB cable between VC709 and monitor Laptop/PC.
  - Note down the new USB tty id. This will be used by the debug connection between the monitor Laptop/PC and the AJIT system on the FPGA.

Preparing and compiling an application

- Write your main application C source/headers in the usual way.
  - Use the print and timer routines provided as part of the AJIT tool-chain.
- Write init.s assembly file to initialize stack for bare-metal execution.
- Prepare a linker script.
- Compile your application using the compile script provided by the AJIT tool-chain.
  - Prepare a compile script to specify assembly files, C source files, include directories, compiler options, defines etc for the script.
  - Run the compile script: this produces a memory map (mmap) file, as well as object dump, elf etc.

Running the application: minicom

- Start minicom on the monitor Laptop/PC.

  `minicom -s`
- Set the TTY id (note: connection to CP210x discussion above).
- Set the baud-rate to 115200.
- Line-feed, carriage return, local echo, log-file as per your requirements.

Running the application: ajit_debug_monitor_mt

- Start ajit_debug_monitor_mt:

  `ajit_debug_monitor_mt -u /dev/ttyUSBx`

  Here the USBx corresponds to the debug connection (mini-B USB to USB) discussed above.
- Go through the following sequence:

  ```
  w rst 1
  m application.mmap
  w rst 0
  ```

  You should be able to see the output of your program in minicom.

Resources

- The `AjitToolChain/AjitPublicResources/ajit_access_routines_` folder contains useful AJIT hardware access routines.
- The `AjitToolChain/AjitPublicResources/minimal_printf_timer` folder contains print and timer routines.

Dhrystone

- Examine linker script.
- Examine code.
- Examine init.s file.
- Examine compile script.
- Run compile script.
- Examine compile results.
- Run the program.

Dhrystone linker script

```
/*=====================================================>
/*                                                     >
/* Linker script for AJIT platform   */
/*                                                     >
/*=====================================================>
ENTRY(_start)
SECTIONS
{
. = 0x0;
.text ALIGN(8) : { *(.text.main) *(.text*) }
. = 0x10000;
.rodata ALIGN(8) : { *(.rodata) }
}
```

Dhrystone init.s file

```
.global main
main:
_start:
set 0xffe0f000, %sp  ! stack
set 0xffe0f000, %fp  ! frame

! mmu control register format
!   [8]                        [7:1]
        !   default-cacheable bit, unused
        !
! mark all accesses as cacheable..
!  (do not use without justification)
!
set 0x100, %o0
sta %o0, [%g0] 0x4

! no arguments..
call ajit_main
```

Dhrystone unified.c file

- Need to supply a timer routine
  - ajit_barebones_clock returns ticks at clock_frequency/256.
- Need to supply a print routine
  - ee_printf.
  - Need to enable serial device.

Compile script

- init.s file (Note: for this example, no trap handlers were supplied).
- unified.c
- AJIT access routines.
- Minimal printf, timer.
- uclibc.

Next steps

- Trap handlers.
- Using GDB with the AJIT single board computer.
- Setting up a virtual to physical memory map while compiling your application.
- Writing and using interrupt service routines.