La distribution XILLINUX sur la carte ZYBO

1) présentation

Xillinux est une distribution de bureau pour LINUX qui permet un environnement graphique de bureau pour la carte ZYBO. Une souris et un clavier peuvent être attachés via le port USB, tandis qu'un moniteur peut être connecté sur le port VGA ou HDMI de la ZYBO. Bien que fournissant une distribution LINUX, Xillybus fournit un kit de développement pour une interaction entre la distribution LINUX et les périphériques commençant sur la logique programmable. Le principal avantage de cette distribution comparée aux autres c'est que le système de fichier racine se trouve sur la partition ext4 de la SDCARD et donc facilement modifiable.

Ceci prend la forme d'une mémoire FIFO coté logique et des opérations sur les fichiers LINUX du coté hôte ou PS.

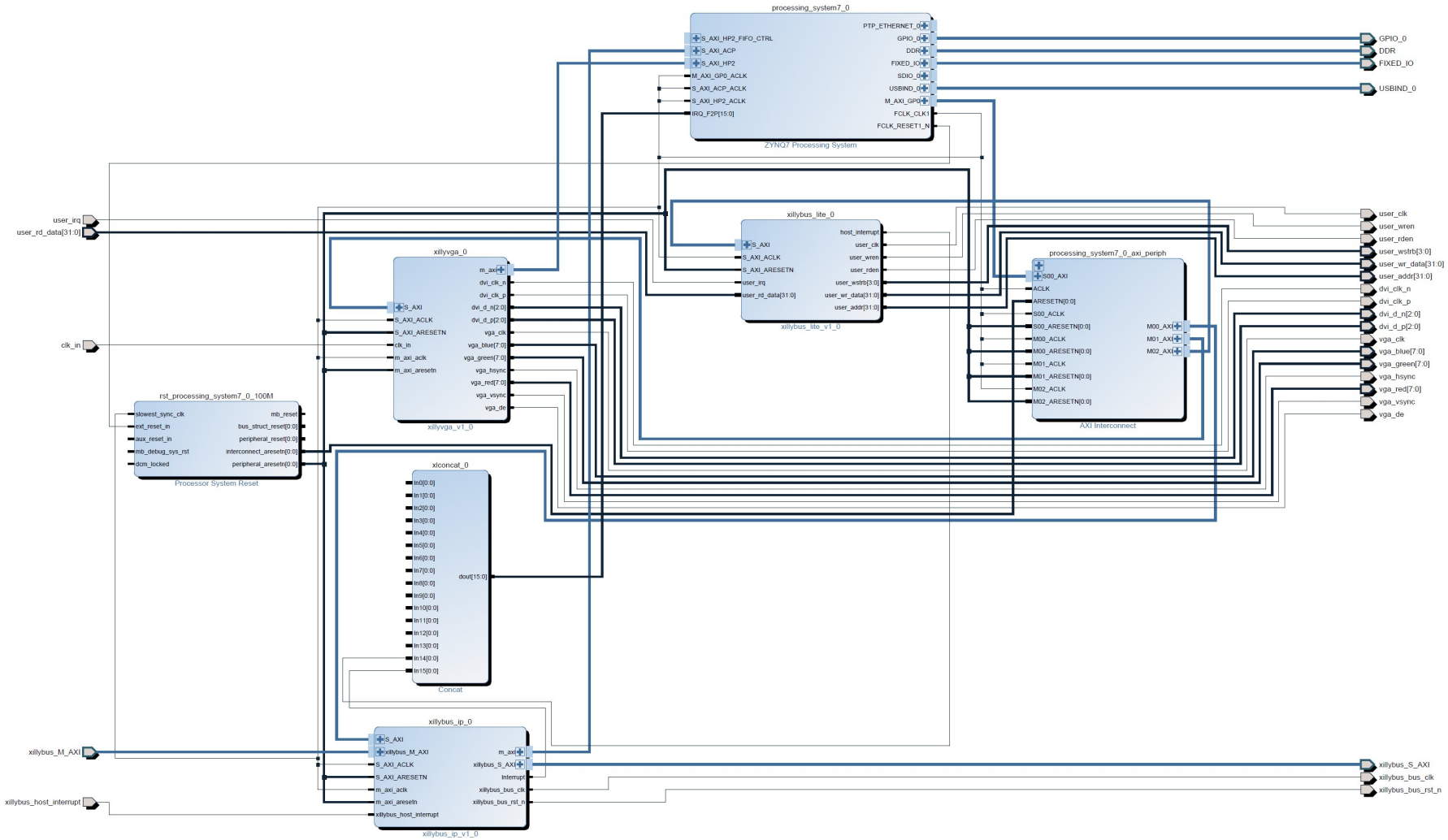
La distribution est basée sur un UBUNTU 12.04 avec un support à long terme (LTS). Les outils logiciels sont disponibles sur le site de Xillybus

2) Implantation matérielle

Récupérez l'image de xillinux sur le site http://xillybus.com/xillinux/
Le fichier http://xillybus.com/downloads/xillinux-eval-zybo-1.3c.zip permet sur vivado de fournir le bitstream pour façonner le FPGA
Ouvrir le projet F:\Xilinx\zybo15_1\ZYBO\xillinux-eval-zybo-1.3c\vivado-essentials\project_1 et ouvrir le bloc design

Si vous avez choisi de travailler en VHDL,
Editez le fichier vhdl/src/xillydemo.vhd afin de mettre en commentaire les lignes suivantes :
PS_CLK : IN std_logic;
PS_PORB : IN std_logic;
PS_SRSTB : IN std_logic;
et décommenter les lignes suivantes dans la définition de l'architecture
-- signal PS_CLK : std_logic;
-- signal PS_PORB : std_logic;
-- signal PS_SRSTB : std_logic;

On peut visualiser le bloc design du projet et faire apparaitre la mémoire FIFO


processing_system7_0
PTP_ETHERNET_0
S_AXI_HP2_FIFO_CTRL
GPIO_0
S_AXI_ACP
DDR
S_AXI_HP2
FIXED_IO
M_AXI_GP0_ACLK
SDIO_0
S_AXI_ACP_ACLK
USBIND_0
S_AXI_HP2_ACLK
M_AXI_GP0
IRQ_F2P[15:0]
FCLK_CLK1
FCLK_RESET1_N
ZYNQ7 Processing System
GPIO_0
DDR
FIXED_IO
USBIND_0
user_irq
user_rd_data[31:0]
clk_in
xillyvga_0
m_axi
dvi_clk_n
dvi_clk_p
S_AXI
dvi_d_n[2:0]
S_AXI_ACLK
dvi_d_p[2:0]
S_AXI_ARESETN
vga_clk
clk_in
vga_blue[7:0]
m_axi_aclk
vga_green[7:0]
m_axi_aresetn
vga_hsync
vga_red[7:0]
vga_vsync
vga_de
xillyvga_v1_0
xillybus_lite_0
S_AXI
host_interrupt
S_AXI_ACLK
user_clk
S_AXI_ARESETN
user_wren
user_irq
user_rden
user_rd_data[31:0]
user_wstrb[3:0]
user_wr_data[31:0]
user_addr[31:0]
xillybus_lite_v1_0
processing_system7_0_axi_periph
S00_AXI
ACLK
ARESETN[0:0]
S00_ACLK
S00_ARESETN[0:0]
M00_AXI
M00_ACLK
M01_AXI
M00_ARESETN[0:0]
M02_AXI
M01_ACLK
M01_ARESETN[0:0]
M02_ACLK
M02_ARESETN[0:0]
AXI Interconnect
user_clk
user_wren
user_rden
user_wstrb[3:0]
user_wr_data[31:0]
user_addr[31:0]
dvi_clk_n
dvi_clk_p
dvi_d_n[2:0]
dvi_d_p[2:0]
vga_clk
vga_blue[7:0]
vga_green[7:0]
vga_hsync
vga_red[7:0]
vga_vsync
vga_de
rst_processing_system7_0_100M
slowest_sync_clk
mb_reset
ext_reset_in
bus_struct_reset[0:0]
aux_reset_in
peripheral_reset[0:0]
mb_debug_sys_rst
interconnect_aresetn[0:0]
dcm_locked
peripheral_aresetn[0:0]
Processor System Reset
xlconcat_0
In0[0:0]
In1[0:0]
In2[0:0]
In3[0:0]
In4[0:0]
In5[0:0]
In6[0:0]
In7[0:0]
dout[15:0]
In8[0:0]
In9[0:0]
In10[0:0]
In11[0:0]
In12[0:0]
In13[0:0]
In14[0:0]
In15[0:0]
Concat
xillybus_ip_0
S_AXI
xillybus_M_AXI
m_axi
S_AXI_ACLK
xillybus_S_AXI
S_AXI_ARESETN
Interrupt
m_axi_aclk
xillybus_bus_clk
m_axi_aresetn
xillybus_bus_rst_n
xillybus_host_interrupt
xillybus_ip_v1_0
xillybus_M_AXI
xillybus_host_interrupt
xillybus_S_AXI
xillybus_bus_clk
xillybus_bus_rst_n

Générez par la suite le fichier bitstream xillydemo.bit pour la SDCARD

3) Création du système de fichier root

Téléchargez le fichier http://xillybus.com/downloads/xillinux-1.3.img.gz qui contient l'image de la SDCARD avec la distribution LINUX
Implantez cette image sur une SDCARD avec un outil de copie binaire type win32diskimager sur windows.

Les deux fichiers boot.bin et devicetree.dtb à partir du répertoire bootfiles fichier xillinux-eval-zybo-1.3c.zip doivent être copiés sur la partition FAT de la SDCARD.
Copier aussi le fichier xillydemo.bit généré précédemment sur la partition FAT de la SDCARD

4) Création du noyau LINUX

récupérer sur la SDCARD dans le système de fichier root le répertoire kernel-patches et copiez ce répertoire sur le cross compilateur ubuntu
cp -Rf kernel-patches/ /usr/src

téléchargez la version du noyau de linux-digilent
git clone https://github.com/Digilent/Linux-Digilent-Dev.git

faire un git pour les patches
cd Linux-Digilent-Dev
git am /usr/src/kernel-patches/00*

Lancez la compilation et les variables d'environnement

PATH=$PATH:/usr/src/u-boot-Digilent-Dev/tools/
make ARCH=arm CROSS_COMPILE=arm-xilinx-linux-gnueabi- UIMAGE_LOADADDR=0x8000 uImage modules -j32

Copiez le noyau uImage sur la SDCARD partition FAT

Les modules sur la SDCARD doivent être installés en adéquation avec le nouveau noyau

make ARCH=arm INSTALL_MOD_PATH=/media/xilinx1/1ab0183b-6dab-4681-b36f-65966b5cb64b/ modules_install

faire un depmod -a pour une prise en compte des modules si nécessaire

5) Implantation de gdbserver Pour le debugage

#add-apt-repository "deb http://archive.ubuntu.com/ubuntu $(lsb_release -sc) main universe"
#sudo apt-get update
#sudo apt-get upgrade
#sudo apt-get update
apt-get install libncurses5-dev
wget http://ftp.gnu.org/gnu/gdb/gdb-7.6.tar.gz
tar -xzf gdb-7.6.tar.gz
cd gdb-7.6/
./configure
make
make install

6) Commande à distance du bureau de la distribution LINUX ubuntu 12.04 LTS

sudo apt-get install x11vnc

lancez le serveur x sur la carte ZYBO

startx

Puis lancez la commande suivante pour le lancement de X11vnc sur la carte ZYBO

sudo apt-get install x11vnc

x11vnc -bg -o %HOME/.x11vnc.log.%VNCDISPLAY -auth /var/run/gdm/auth-for-gdm*/database -display :0

Connectez vous sur windows avec vncviewer par exemple