Sistema de adquisición basado en FPGA

Revisión 4: freeRTOS + LWIP

Sistema de adquisición basado en FPGA

Arquitectura del sistema inicial

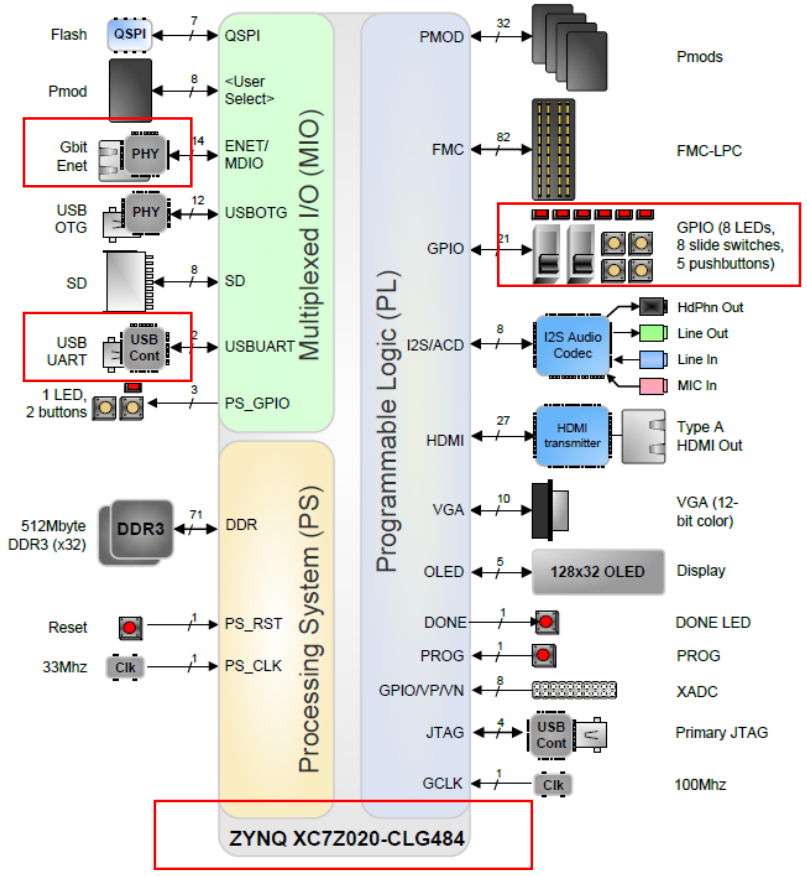


Figure 1 – ZedBoard Block Diagram

El objetivo de este documento es aclarar las especificaciones de la FPGA, la librería LWIP encargada de realizar los protocolos TCP/IP sin sistema operativo y el intercambio de variables entre el main y las interrupciones.

Los bloques de la Zedboard que utilizaremos para las pruebas son los destacados en rojo.

Sistema de adquisición basado en FPGA

Zynq 7000 SoC

El integrado Z-7020 basado en la arquitectura Zynq 7000 SoC de Xilinx cuenta con 4.9Mb de BRAM.

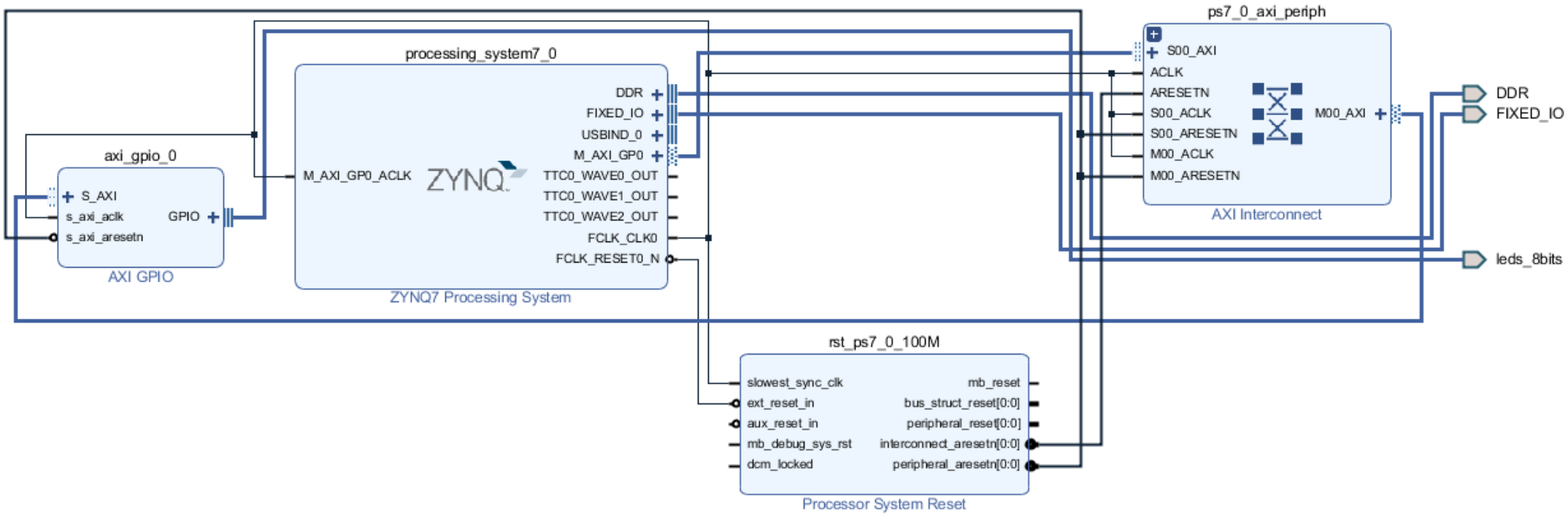
El bloque de sistemas utilizado para configurar el hardware se muestra en la siguiente diapositiva:

Table 1: **Zynq-7000 and Zynq-7000S SoCs *(Cont'd)***

	Device Name	Z-7007S	Z-7012S	Z-7014S	Z-7010	Z-7015	Z-7020	Z-7030	Z-7035	Z-7045	Z-7100
	Part Number	**XC7Z007S**	**XC7Z012S**	**XC7Z014S**	**XC7Z010**	**XC7Z015**	**XC7Z020**	**XC7Z030**	**XC7Z035**	**XC7Z045**	**XC7Z100**
Programmable Logic	Xilinx 7 Series Programmable Logic Equivalent	Artix®-7 FPGA	Artix-7 FPGA	Artix-7 FPGA	Artix-7 FPGA	Artix-7 FPGA	Artix-7 FPGA	Kintex®-7 FPGA	Kintex-7 FPGA	Kintex-7 FPGA	Kintex-7 FPGA
	Programmable Logic Cells	23K	55K	65K	28K	74K	85K	125K	275K	350K	444K
	Look-Up Tables (LUTs)	14,400	34,400	40,600	17,600	46,200	53,200	78,600	171,900	218,600	277,400
	Flip-Flops	28,800	68,800	81,200	35,200	92,400	106,400	157,200	343,800	437,200	554,800
	Block RAM (# 36 Kb Blocks)	1.8 Mb (50)	2.5 Mb (72)	3.8 Mb (107)	2.1 Mb (60)	3.3 Mb (95)	4.9 Mb (140)	9.3 Mb (265)	17.6 Mb (500)	19.2 Mb (545)	26.5 Mb (755)
	DSP Slices (18x25 MACCs)	66	120	170	80	160	220	400	900	900	2,020
	Peak DSP Performance (Symmetric FIR)	73 GMACs	131 GMACs	187 GMACs	100 GMACs	200 GMACs	276 GMACs	593 GMACs	1,334 GMACs	1,334 GMACs	2,622 GMACs
	PCI Express (Root Complex or Endpoint)[3]		Gen2 x4			Gen2 x4		Gen2 x4	Gen2 x8	Gen2 x8	Gen2 x8
	Analog Mixed Signal (AMS) / XADC	2x 12 bit, MSPS ADCs with up to 17 Differential Inputs									
	Security[2]	AES and SHA 256b for Boot Code and Programmable Logic Configuration, Decryption, and Authentication									

Sistema de adquisición basado en FPGA

Configuración hardware



Sistema de adquisición basado en FPGA

Aplicaciones embebidas en Zynq 7000 SoC

Una vez generada la configuración para el hardware mediante el kit de desarrollo de software de Xilinx se diseñaron cuatro versiones del código para realizar las siguientes pruebas:

Versión 26: Crear red con cliente en Matlab y configurar librería LWIP.

Versión 27_3: Buscar longitud máxima de envío desde el servidor al cliente.

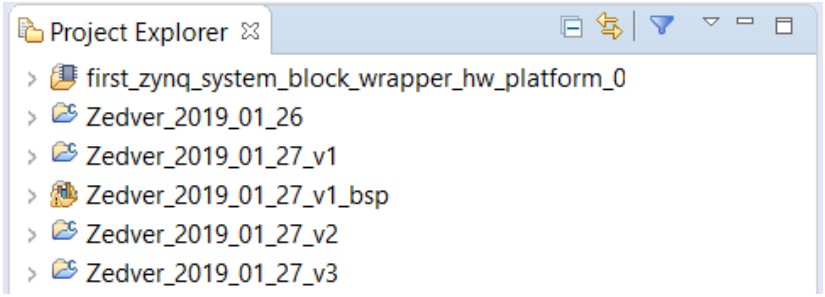


Figura 1

Figura 2

Sistema de adquisición basado en FPGA

Versión 26

Partiremos de la plantilla de aplicación servidor en LWIP, sin usar sistema operativo, modificaremos las opciones de la librería para desactivar el protocolo DHCP.

Añadimos el control del puerto GPIO en el bucle infinito del procesador en el main y comprobamos que la función de recepción en el servidor funciona como una interrupción, al ser ejecutada vuelve al bucle infinito.

El intercambio de variables entre el servidor y el main, al encontrarse en diferentes scripts, se realizará mediante variables extern en el script del servidor.

Sistema de adquisición basado en FPGA

Versión 26: Main

```
int main(void)
{
    //Leds:
    int Status;
    int led = 0; /* Hold current LED value. Initialise to LED definition */
    Status = XGpio_Initialize(&Gpio, GPIO_DEVICE_ID);
    XGpio_SetDataDirection(&Gpio, LED_CHANNEL, 0x00);
    XGpio_DiscreteWrite(&Gpio, LED_CHANNEL, led);
```

Apagamos leds

```
 while (1) {
     if (TcpFastTmrFlag) {
         tcp_fasttmr();
         TcpFastTmrFlag = 0;
     }
     if (TcpSlowTmrFlag) {
         tcp_slowtmr();
         TcpSlowTmrFlag = 0;
     }
     xemacif_input(netif);

     //Server receive request
     if (flag_received == 1){
         flag_received = 0;
         led++;
         XGpio_DiscreteWrite(&Gpio, LED_CHANNEL, led);
         xil_printf("Received at main:%u\n\r", received);
     }
```

Incrementamos valor binario de los leds y actualizamos su valor en el GPIO

Sistema de adquisición basado en FPGA

Versión 26: Server

```
//Request declarations
extern int flag_received;
extern int received;
int *received_pointer;
```

Interrupción para recibo TCP en nivel superior

```
/** Receive data on a tcp session */
static err_t tcp_recv_perf_traffic(void *arg, struct tcp_pcb *tpcb,
        struct pbuf *p, err_t err)

  tcp_recved(tpcb, p->tot_len);

  //Server variable management
  flag_received = 1;
  received_pointer = p->payload;
  xil_printf("Received:%u\n\r", *received_pointer);
  received = *received_pointer;

  pbuf_free(p);
  return ERR_OK;
```

Bandera avisa al main de que hay dato recibido y lo extrae del puntero usado por TCP en una variable que también tiene acceso el main.

Sistema de adquisición basado en FPGA

Versión 27

En la interrupción de recepción en TCP añadiremos la función de escribir para contestar a las peticiones de entrada.

Tras probar que los valores recibidos son los correctos, se intento con un vector de 10 posiciones, al ver que era posible se incremento hasta observar que valores muy elevados provocaban que el cliente no recibiese el mismo numero que enviados, aunque estos se queden en cola. El valor máximo encontrado de posiciones del vector enviado de variables uint32 es de 16000, pudiendo enviar 512kB con una única llamada a la función de escritura.

Sistema de adquisición basado en FPGA

Versión 27: Main y cliente Matlab

```
#define BUF_SIZE 16000

//Server declarations
uint32_t flag_received = 0;
uint32_t received = 0;
uint32_t sended;
uint32_t send_buf [BUF_SIZE];
int main(void)
{
    for (int i = 0; i<BUF_SIZE; i++)
        send_buf[i]=0xFFFFFFFF;
```

```
%% Inicailización
clc, clear all, close all
puertoEcho = 7;
puertoServer = 5001;
puertoLocal = 30000;
ipZed = '192.168.1.10';
ipLocal = 'localhost';
%% Cliente
t = tcpclient (ipZed,puertoServer);
data = cast(hex2dec('0000'),'uint32');
%% Envio

write(t,data)
%% Envio 2

a = read(t);
```

Sistema de adquisición basado en FPGA

Versión 27: Servidor

```
#define BUF_SIZE 16000

extern struct netif server_netif;
static struct tcp_pcb *c_pcb;
static struct perf_stats server;

//Request declarations
extern uint32_t flag_received;
extern uint32_t received;
extern uint32_t sended;

extern uint32_t send_buf [BUF_SIZE];

uint32_t *received_pointer;
uint32_t *sended_pointer;
```

```
//----------------------------------------
//-------Server variable management--------
//----------------------------------------
    //Received
    flag_received = 1;
    received_pointer = p->payload;
    received = *received_pointer;
    xil_printf("Received:%u\n\r", received);
    //Sending
    sended = 0xFFFFFFFF;
    if(received == 0){
    sended_pointer = &send_buf;
    p->payload = sended_pointer;
    err = tcp_write(tpcb, p->payload, sizeof(send_buf), 1);
    }
    else{
    sended_pointer = &sended;
    p->payload = sended_pointer;
    err = tcp_write(tpcb, p->payload, sizeof(send_buf), 1);
    }
```

Sistema de adquisición basado en FPGA

Conclusiones

- Al terminar la interrupción de recepción del servidor, el procesador vuelve a ejecutar el bucle infinito del main.
- Matlab recibe separando los bytes del uint32 en 4 uint8.
- Teniendo en cuenta que la memoria BRAM disponible es de 4900kb que equivalen a 612kB no podremos ocupar 512kB para enviar máximo de la función de escritura.
- Al ejecutar una aplicación que no funcione puede provocar que no funcionen otras aplicaciones y la solución temporal es reiniciar el ordenador y el hardware de la FPGA.