INFORME DE PROYECTO - CONTROL DE VELOCIDAD DE AUTO

Patricio Germán Silva*

Introducción a VHDL - Ingeniería en Mecatrónica

30 de noviembre de 2024

*Correo electrónico: silvap@fcal.uner.edu.ar

Índice

1. Introducción

Se realizará el diseño, descripción en VHDL y simulación de un sistema capaz de realizar el control de un auto bimotor seguidor de linea. El diseño debe contemplar:

- Control de dirección y velocidad para un driver L293D.
- Control de un display de siete segmentos donde se muestra información de velocidad y modo de funcionamiento.
- Comunicación mediante interfaz UART y protocolo rs232 (TTL) con una velocidad de 9600 baudios, 8 bits, de datos, 1 bit de stop, sin paridad, sin control de flujo. Solo se requiere recepción.
- Para la comunicación se agrega una capa de protocolo, de frame de ancho fijo, compuesto por un byte de header, un byte de comando, dos bytes de datos y un byte de tráiler, sin suma de comprobación.

Para el desarrollo se utiliza el software **Xilinx Vivado 2024.1**. Por ultimo, se comprobará el correcto funcionamiento del desarrollo en una placa **DIGILENT Artix-7 FPGA modelo Arty A7-100T (xc7a100tcsg324-1)**.

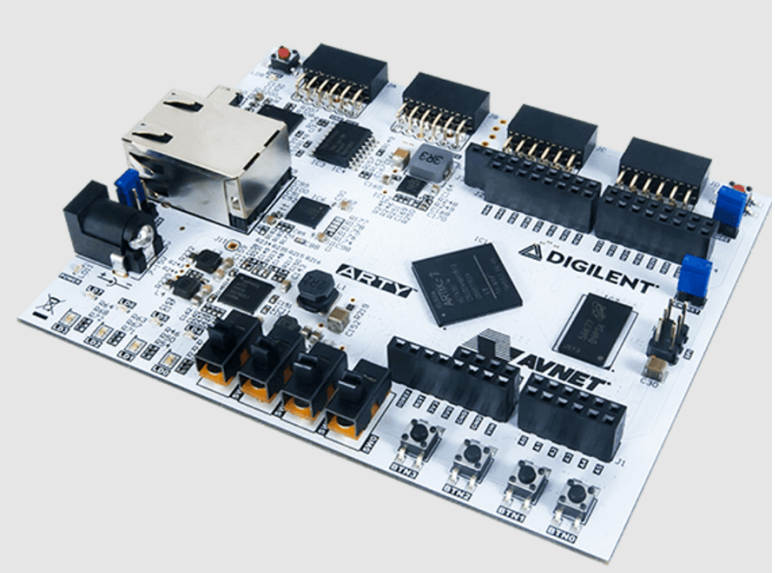


Figura 1: DIGILENT Artix-7 FPGA modelo Arty A7-100T

Los comandos son los siguientes:

COMANDO	Descripción
D000Z	STOP
D1nnZ	Velocidad Motor Derecho. nn = 10 a 90 en pasos de a 5.
D2nnZ	Velocidad Motor Izquierdo. nn = 10 a 90 en pasos de a 5.
D3nnZ	Selecciona la velocidad media para recorrer la pista. nn = 10 a 90 en pasos de a 5.
D4sxZ	Simulador de sensores. s: Simula Sensores encendidos. x: Cualquier valor.
D5sxZ	Selecciona modo de control. s: 0 Control desde PC s: 1 Control con sensores. x: Cualquier valor

2. Diseño de la solución

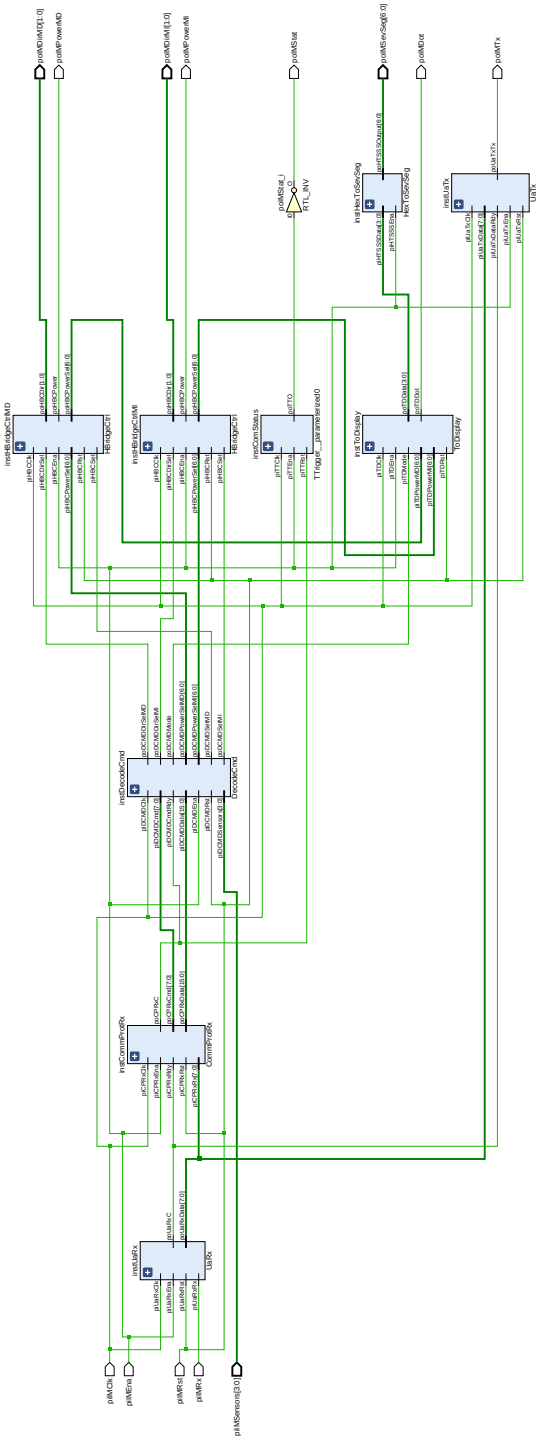
El diseño se lleva a cabo en diferentes módulos específicos para cada tarea, que serán los siguientes:

- UaRx: modulo Uart RX para la recepción desde la PC
- UaTx: modulo Uart TX, cuya única finalidad es hacer echo de la información recibida por UaRx.
- CommProtRx: modulo que valida un paquete del protocolo de comunicación, e incorpora una detección de timeout.
- DecodeCmd: Interpreta los comandos recibidos.
- HBridgeCtrl: recibe una dirección y velocidad y genera una salida PWM según los parámetros establecidos.
- ToDisplay: Envia información a un display de 7 segmentos con punto decimal
- Módulos auxiliares: módulos contadores, generadores de baud rate y PWM y decodificadores de 7 segmentos.

2.1. Puertos de I/O

NOMBRE	DIRECCION	PIN	HEADER	DESCRIPCION
piIMClk	IN	E3	-	Clock de sistema 100MHz
piIMEna	IN	A8	SW0	Enable
piIMRst	IN	C11	SW1	Reset
piIMRx	IN	A9	-	Salida al puerto TX del PC
piIMSensors[0]	IN	B8	BTN0	Simula sensor externo izquierdo
piIMSensors[1]	IN	B9	BTN1	Simula sensor interno izquierdo
piIMSensors[2]	IN	C9	BTN2	Simula sensor interno derecho
piIMSensors[3]	IN	D9	BTN3	Simula sensor externo derecho
poIMDirMD[0]	OUT	J5	LED4	Dir1 – Segun L243D
poIMDirMD[1]	OUT	H5	LED5	
poIMDirMI[0]	OUT	T10	LED6	Dir2 – Segun L243D
poIMDirMI[1]	OUT	T9	LED7	
poIMDot	OUT	U11	IO26	Punto decimal del display de 7 segmentos
poIMPowerMD	OUT	N17	IO40	Salida PWM derecho – 100Hz
poIMPowerMI	OUT	P18	IO41	Salida PWM izquierdo – 100Hz
poIMSevSeg[0]	OUT	V16	IO33	Display – Segmento A
poIMSevSeg[1]	OUT	M13	IO32	Display – Segmento B
poIMSevSeg[2]	OUT	R10	IO31	Display – Segmento C
poIMSevSeg[3]	OUT	R11	IO30	Display – Segmento D
poIMSevSeg[4]	OUT	R13	IO29	Display – Segmento E
poIMSevSeg[5]	OUT	R15	IO28	Display – Segmento F
poIMSevSeg[6]	OUT	P15	IO27	Display – Segmento G
poIMStat	OUT	G6	LED0_R	200ms blink en cada comando valido
poIMTx	OUT	D10	-	Salida al puerto RX del PC

2.2. Esquemático general del diseño es el siguiente



3. Módulos

A continuación se describen los módulos desarrollados

3.1. UaRx

Formado por una maquina de estado y un generador de baudrate. EL puerto RX conectado al uart recibe desde la pc paquetes de datos de 8 bits a 9600 baudios, sin control de paridad ni control de flujo, con 1 bit de stop. Cuando un nuevo dato se recibe se genera un pulso de 1 clock de duración que notifica el evento.

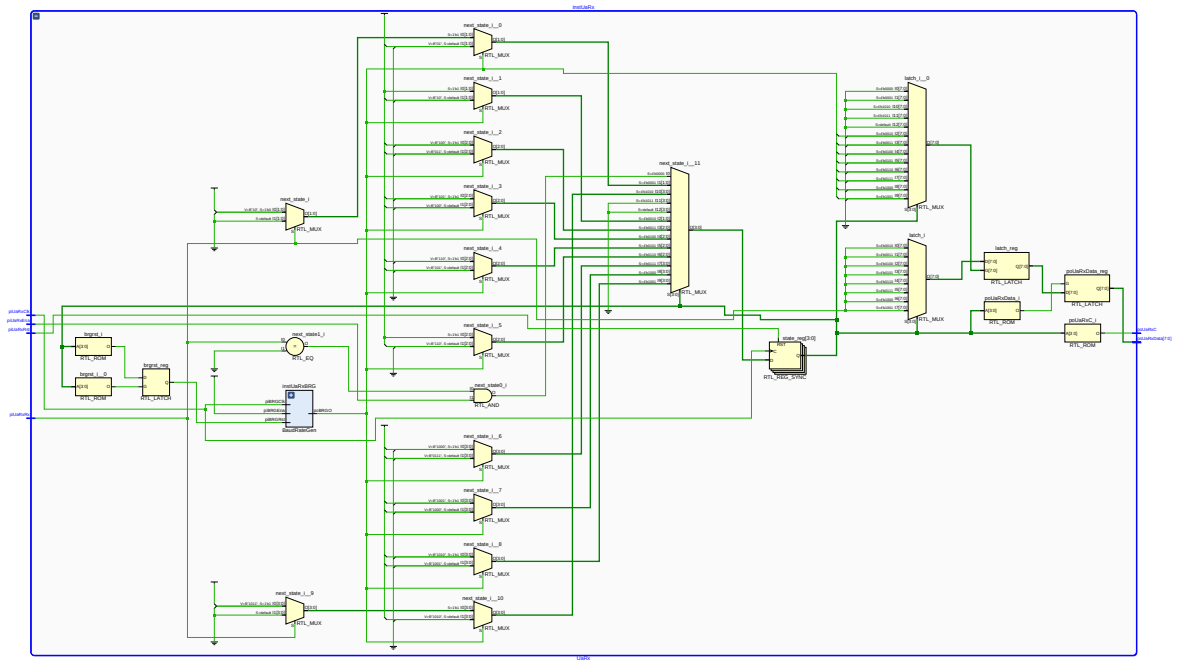


Figura 2: Esquemático del modulo UaRx

3.2. UaTx

3.3. CommProtRx

3.4. DecodeCmd

3.5. HBridgeCtrl

3.6. ToDisplay

3.7. Módulos auxiliares

3.7.1. ModuleCounter

3.7.2. Counter

3.7.3. BaudRateGen

3.7.4. TTrigger

3.7.5. PwmGen

3.7.6. HexToSevSeg

4. Conclusiones

Nos resulta imposible no comparar esta tarea con la asignada el año pasado en Robotica I. La tarea en si no es mucho mas compleja sin embargo el desarrollo de esta consigna requirió un enfoque más orientado hacia la flexibilidad del programa. El resultado es un programa con mucho menos lineas que el del año pasado que realiza la tarea de tomar una cantidad arbitraria de cajas desde una estantería de cualquier tamaño y forma ubicada en cualquier lugar de la zona de trabajo del robot y las coloque en cualquier otra posición de la zona de trabajo en forma de pirámide.

Los desafíos que esto nos trajo nos hace ver que las posibilidades son muy amplias y que resta mucho por aprender, la aplicación de las buenas practicas es algo fundamental para el desarrollo de soluciones adecuadas y flexibles pero la información en un área especifica como la programación de robots manipuladores no es muy abundante, lo que obliga a investigar de forma proactiva modos alternativos de realizar las tareas y así ir seleccionando aquellas metodologías que mejor se ajustan a cada situación.