DESARROLLO DE UN GUANTE BASADO EN ACELERÓMETROS PARA DETERMINAR LA POSTURA DE LA MANO A PARTIR DE LA MEDICIÓN DE LOS ÁNGULOS EN LAS ARTICULACIONES

DUVERNEY CORRALES MENDOZA

UNIVERSIDAD DEL VALLE
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA
PROGRAMA ACADÉMICO DE INGENIERÍA ELECTRÓNICA
SANTIAGO DE CALI, 2015

DESARROLLO DE UN GUANTE BASADO EN ACELERÓMETROS PARA DETERMINAR LA POSTURA DE LA MANO A PARTIR DE LA MEDICIÓN DE LOS ÁNGULOS EN LAS ARTICULACIONES

Estudiante:

DUVERNEY CORRALES MENDOZA

Trabajo de grado para optar al título de Ingeniero Electrónico

Director:

MARIO VERA LIZCANO

UNIVERSIDAD DEL VALLE
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA
PROGRAMA ACADÉMICO DE INGENIERÍA ELECTRÓNICA
SANTIAGO DE CALI, 2015

DEDICATORIA

Dedico este trabajo de grado a eso que yo llamo Dios, por cada lagartija, mariposa, pajarito y demás recursos naturales que ha usado para dirigirme y corregirme. Por inspirarme cuando más lo necesito y por cuidarme. Por ayudarme a superar cada crisis y obstáculo que se me presenta.

AGRADECIMIENTOS

Primero que todo, agradezco a mis padres por apoyarme en mi sueño de ser ingeniero y profesor universitario. Sin su ayuda, lo más probable es que no hubiese estudiado la ingeniería y no estuviese encaminado en este proceso. También agradezco a mi hermanita por su compañía en Cali durante los últimos años. Últimamente he aprendido a valorar mucho la compañía de las personas a las que quiero.

Agradezco al profe Mario, por la confianza y el apoyo que me ha dado durante estos años. Porque no solo es mi director de trabajo de grado sino también mi amigo; Por enseñarme a ver el mundo de una forma más global y por enseñarme a pensar de una forma más profesional. De su visión nació la idea de este proyecto y de su motivación, los deseos de emprender en esta bonita y constructiva experiencia.

Quiero agradecer a Adolfo (Fito) porque fue importante para mí recibir su opinión y consejos en un momento en que realmente los necesitaba, fue bastante oportuno, y también a juanita por su amistad y compañía.

Agradezco al profe Eduardo Marlés, con quien tuve la oportunidad de trabajar a través de una monitoria, por tres cosas en especial: por la frase “lo excelente es enemigo de lo bueno”, por enseñarme que se debe servir a los demás y no solo a sí mismo, y por presentarme la herramienta computacional Mathematica, la cual me encanta, como lo demuestro en este trabajo.

Finalmente, agradezco a mis compañeros y profesores de la Escuela de Ingeniería Eléctrica y Electrónica, por acompañarme durante este proceso. Algunos, incluso, me hicieron pasar momentos felices.

Duverney Corrales Mendoza

Contenido

Capítulo 1 Generalidades 8
1.1. Estructura cinemática de la mano 9
1.1.1. Huesos de la mano 9
1.1.2. Articulaciones de la mano 10
1.2. Acelerómetro ADXL345 11
1.3. Ángulo entre 2 planos 11
1.3.1. Ángulo entre 2 planos a partir de su inclinación 12
1.3.2. Coordenadas esféricas 13
1.3.3. Método de Kurata 14
1.4. Objetivos 16
1.5. Mapa del proyecto 16
1.6. Quartus II 17
1.6.1. RTL Viewer 17
1.6.2. SignalTap 17
1.7. Wofram Mathematica 18
1.7.1. Comunicación con dispositivos externos 18
1.7.2. Ejecución de procesos síncronos 19
1.7.3. Visualización dinámica de la información 19
1.8. Referencias 20
Capítulo 2 Centralización de los datos 22
2.1. Bus SPI 22
2.2. Interfaces basadas en SPI 23
2.2.1. SPI de 3 hilos 23
2.2.2. Múltiples interfaces SPI con conexión dedicada 24
2.2.3. Configuración Daisy-Chain 25
2.2.4. mSPI (mini-SPI) 25
2.2.5. Multi I/O SPI 26
2.3. Interfaz multi I/O SPI para la centralización de datos de 17 acelerómetros ADXL345. 27
2.3.1. Unidad de control 27

2.3.2. Datapath de interfaz SPI 29
2.3.3. Registro de desplazamiento 30
2.3.4. Datapath de Interfaz Multi I/O SPI 30
2.3.5. Registro de almacenamiento paralelo 30
2.4. Corrección de Offset 31
2.4.1. Implementación de sistema de corrección de offset. Sugerencia del fabricante 31
2.4.2. Implementación de sistema de corrección de offset. Procedimiento experimental. 32
2.5. Resultados 33
2.6. Referencias 33
Capítulo 3 Calculo de los ángulos de las articulaciones 35
3.1. Caso 1. Cálculo de los ángulos con Mathematica. 35
3.1.1. Trama transmitida 36
3.1.2. Adquisición y organización de los datos 36
3.1.3. Cálculo de ángulos mediante coordenadas esféricas 38
3.1.4. Cálculo de ángulos mediante el método de kurata 38
3.2. Caso 2. Cálculo de ángulos en la FPGA. 39
3.2.1. Diseño e implementación de la función Arco-Tangente 39
3.2.2. Cálculo de los ángulos presentes en las articulaciones 44
3.2.3. Trama transmitida 45
3.2.4. Adquisición y organización de los datos en Mathematica 45
3.3. Análisis de casos: Procesamiento Externo vs Local 46
3.4. Recursos de hardware utilizados 47
3.5. Representación visual de la mano 47
3.5.1. Modelo de la mano 48
3.5.2. Creación de los segmentos de la mano con objetos virtuales 3D 48
3.5.3. Ensamble de los segmentos 3D y asociación con los ángulos medidos 49
3.5.4. Visualización dinámica de la postura de la mano 50
3.6. Referencias 51
Capítulo 4 Construcción del DataGlove 52
4.1. Aspectos Técnicos de los Componentes Utilizados 52
4.1.1. Acelerómetro ADXL345 52
4.1.2. Módulos XBEE 54
4.1.3. Tarjeta de desarrollo DE0-nano 54

4.2. Fabricación del Guante Electrónico 56
4.2.1. Fabricación del guante textil 56
4.2.2. Instalación del sistema electrónico 57
4.2.3. Acople del sistema electrónico con el guante textil 59
4.2.4. Costos 59
4.3. Procesos de Verificación 60
4.3.1. Verificación a nivel eléctrico 60
4.3.2. Verificación de la comunicación UART 60
4.3.3. Verificación a nivel lógico 61
4.3.4. Verificación a nivel de datos 61
4.3.5. Verificación de la medición de ángulos 62
4.3.6. Comparación con plataformas comerciales similares 62
4.4. Referencias 63
Capítulo 5 Conclusiones 64
Capítulo 6 Recomendaciones para trabajos futuros 65

Capítulo 1
Generalidades

El uso de la postura de la mano es una forma natural e intuitiva de interacción y comunicación humano-humano [1], pudiendo llegar a serlo también para la interacción humano-máquinas. El reconocimiento de la postura de la mano es útil en diversas áreas: robótica, salud, electrónica, mecánica, etc.; en las cuales se han desarrollado proyectos de interacción robot-humano (human-robot interaction HRI), interfaz entre humanos y dispositivos computacionales, sistemas de diagnóstico para rehabilitación [2], telemedicina, análisis de la cinemática de la mano durante la actividad física deportiva, manipuladores remotos, tele-navegación, realidad virtual, juegos de video, entre otras.

En los últimos veinte años se han desarrollado guantes electrónicos, en su mayoría diseñados para aplicaciones puntales: remplazo de teclados y ratones, controles de juegos, sistema para lenguaje de señas, sistemas para rehabilitación de la mano, sistemas de entrenamiento industrial, sistemas tele-operados, etc. [3]. Los sensores tradicionalmente utilizados en estos guantes electrónicos son: interruptores, flexómetros, giroscopios, acelerómetros con interfaz análoga, acelerómetros con interfaz digital, IMUs, etc.

La medición de variables cinemáticas puede realizarse considerando como vectores de referencia dos fenómenos naturales: la gravedad y el campo magnético. La gravedad como vector de referencia vertical, apuntando siempre al centro de la Tierra, mientras que el campo magnético como vector de referencia horizontal, apuntando siempre al Norte.

En este proyecto se demuestra que la medición y procesamiento de la fuerza gravitacional permite determinar la postura de la mano. Para lo cual se construyó un sistema basado en acelerómetros, que asocia los valores registrados con los vectores normales a la superficie dorsal de los segmentos de la mano.

El sistema desarrollado integra tecnologías modernas. Para la adquisición de la gravedad, acelerómetros tipo MEMS; Para la centralización y pre-procesamiento de datos, una FPGA; Para la comunicación al PC, un sistema RF; y para el post-procesamiento y visualización, Wolfram Mathematica.

1.1. *Estructura cinemática de la mano*

1.1.1. *Huesos de la mano*

La mano está compuesta por veintisiete huesos: catorce falanges en los dedos y en la palma cinco metacarpianos y ocho carpianos (ver **Figura 1.1**). Los dedos tienen tres falanges: falange distal, falange media y falange proximal, mientras que el dedo pulgar tiene dos falanges. Los metacarpianos se ubican entre las falanges proximales y los huesos carpianos, ocupando casi toda la palma de la mano. Los carpianos hacen parte de la muñeca (unión entre el antebrazo y la mano), y están distribuidos en dos hileras. En la hilera distal se encuentra el ganchoso, el grande, el trapecio y el trapezoide, mientras que en la hilera proximal se encuentran el pisiforme, el piramidal, el semilunar y el escafoides.

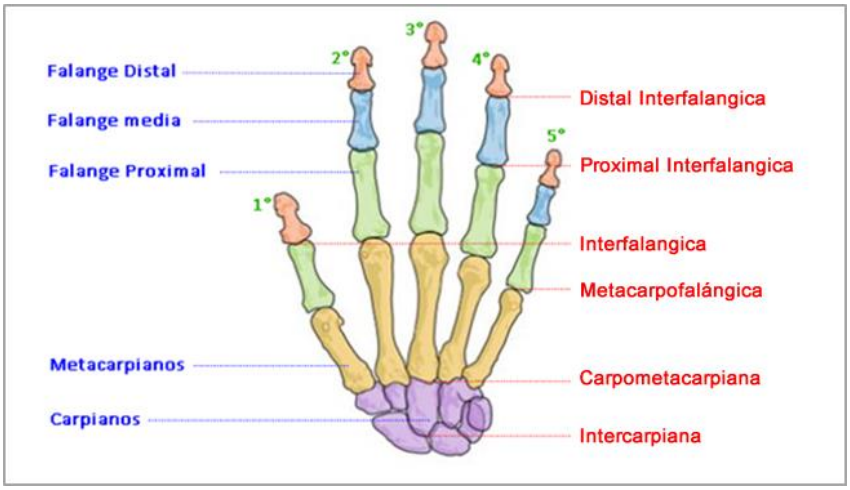


Figura 1.1 Huesos y articulaciones de la mano.

Algunos estudios indican que existe una relación porcentual entre el tamaño de los huesos de la mano y su dimensión total [4], como se indica en la **Tabla 1.1**, mientras que otros estudios proponen que las longitudes de las falanges y metacarpianos de cada dedo, tomadas desde los ejes de rotación, están relacionados mediante "la sucesión de Fibonacci" [5], como se indica en la **Figura 1.2**.

Dedo \ Falange	Proximal	Medial	Distal
Pulgar	17.1	-	12.1
Índice	21.8	14.1	8.6
Corazón	24.5	15.8	9.8
Anular	22.2	15.3	9.7
Meñique	17.2	10.8	8.6

Tabla 1.1 Relación porcentual de las dimensiones de las falanges con respecto a la mano

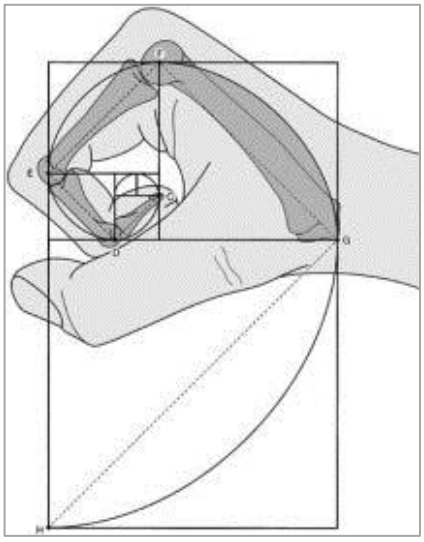


Figura 1.2 Relación de la mano humana con las secuencias de Fibonacci

1.1.2. *Articulaciones de la mano*

La mano tiene cuatro grupos de articulaciones: interfalángicas (AI), metacarpofalángicas (MCF), carpometacarpianas (CMC) e intercarpianas (IC). En la **Tabla 1.2** se presenta una lista de los rangos promedio de movilidad presentes en las articulaciones de la mano [4].

ARTICULACIÓN			RDM (°)
Muñeca		Flexión	0 – 75
		Extensión	0 – 70
		Radial	0 – 20
		Cubital	0 – 35
Pulgar	Carpometacarpiana (CMC)	Aducción Palmar	0
		Abducción Palmar	0 – 45
		Aducción Radial	0
		Abducción Radial	0 – 60
	Interfalángica (AI)	Hiperextensión	0 – 15
		Flexión	0 – 80
	Metacarpofalángica (MCF)	Hiperextensión	0 – 10
		Flexión	0 – 55
Articulación Distal Interfalángica (DIF)		Extensión	0
		Flexión	0 – 80
Articulación Proximal Interfalángica (PIF)		Extensión	0
		Flexión	0 – 10
Articulación Metacarpofalángica (MCF)		Hiperextensión	0 – 45
		Flexión	0 – 90

Tabla 1.2 Rango de movilidad de las articulaciones presentes en la mano. (RDM = Rango de Movilidad)

Las AI unen internamente los huesos de los dedos. Como el pulgar no tiene una falange media, es suficiente referirse a su articulación interna como AI; el resto de los dedos tienen una articulación distal interfalángica (DIF) y una articulación proximal interfalángica (PIF), las cuales están presentes en las uniones de las falanges distal-media y media-proximal, respectivamente.

Los dedos se unen a la palma a través de las articulaciones MCF, las cuales están presentes entre la falange proximal y el metacarpiano de cada dedo. A su vez, los metacarpianos se unen a los carpianos a través de las articulaciones CMC, y las articulaciones IC enlazan los huesos carpianos entre sí. La unión entre la mano y el antebrazo se llama muñeca. En la **Figura 1.1** se presenta como están distribuidos los huesos y las articulaciones de la mano.

1.2. *Acelerómetro ADXL345*

El acelerómetro ADXL345 es un dispositivo MEMS de tres ejes, capaz de medir la aceleración estática de la gravedad (para usarse como sensor de inclinación), así como la aceleración dinámica resultante de movimientos, choques y vibraciones [6].

Los acelerómetros fueron creados en la segunda década del siglo XX y desde entonces han evolucionado notablemente. Años atrás era común encontrar acelerómetros grandes y pesados, analógicos, costosos y con baja eficiencia energética. Hoy en día, gracias a los avances en **M**icroelectrónica y **N**anoelectrónica, es posible fabricar **S**istemas **E**lectro**M**ecánicos de tamaño muy pequeño, a los que se conoce como dispositivos **M**EMS y **N**EMS, Los cuales han permitiendo integrar en un chip no solo los sensores MEMS y NEMS, sino todos o gran parte de los componentes de un sistema electrónico, por lo que es posible encontrar en el mercado componenetes SoC (System on Chip) que incluyen tecnologías MEMS y NEMS. Esto se refleja en productos muy eficientes energéticamente, livianos, compactos, diminutos, económicos, con alto desempeño y con todos los beneficios de los sistemas digitales.

1.3. *Ángulo entre 2 planos*

Calcular el ángulo entre 2 planos es equivalente a calcular el ángulo entre sus vectores normales, como se indica en la **Figura 1.3**. Existen varias técnicas matemáticas [7]: producto punto (Coseno), producto vectorial (Seno) y producto-vectorial/producto-punto (Tangente), los cuales se consignan en la **Tabla 1.3**. Los ángulos obtenidos con estas ecuaciones corresponden a la menor distancia angular presente entre dichos planos, mas no se puede determinar el signo del ángulo debido a que las rectas generadas por la intercepción de los planos pueden dirigirse en cualquier dirección, lo que impide establecer un marco de referencia.

TÉCNICA	EXPRESIÓN
Producto Punto	$\emptyset = \text{ArcCos}\left[\frac{A.B}{\text{Norm}[A]\ \text{Norm}[B]}\right]$
Producto Vectorial	$\emptyset = \lvert\text{ArcSin}\left[\frac{\text{Norm}[A \times B]}{\text{Norm}[A]\ \text{Norm}[B]}\right]$
Producto Punto . Producto Vectorial	$\emptyset = \text{ArcTan}\left[\frac{\text{Norm}[A \times B]}{A.B}\right]$

Tabla 1.3 Funciones matemáticas para hallar el ángulo entre 2 vectores.

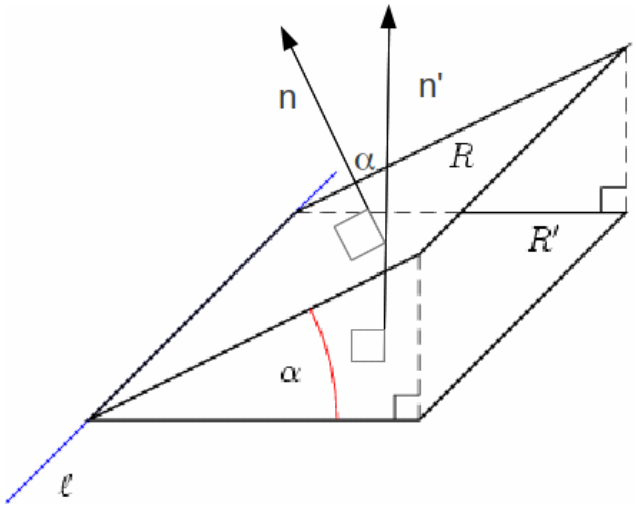


Figura 1.3 Ángulo entre 2 planos.

1.3.1. *Ángulo entre 2 planos a partir de su inclinación*

La inclinación está muy relacionada con el vector normal al plano, pero está referenciada al plano de medición, no al marco de referencia donde se encuentra dicho plano, lo cual se refleja en que no considera rotaciones en torno al eje Z. Por ejemplo, un edificio puede estar inclinado dos grados a la derecha, donde la derecha puede ser el Norte, el Oriente, etc.

En baja frecuencia, la medición del acelerómetro se debe prácticamente a la fuerza gravitacional, cuya medición es idealmente una constante. Esto implica que todas las mediciones en los tres ejes se pueden mapear sobre la superficie de una esfera [7] cuyo radio es proporcional a la magnitud de la gravedad.

Si dicho acelerómetro se hace girar sobre uno de sus ejes, la medición resultante es ubicada sobre la circunferencia formada por la intercepción entre la superficie de la esfera y un cono asociado a la inclinación del eje de rotación. Mientras más horizontal esté el eje de rotación, más grande es la circunferencia resultante (**Figura 1.4**) lo que equivale a una mejor resolución en la medición. Cuando la inclinación se aproxima a los 90°, el radio de la circunferencia tiende a cero, por lo que es más difícil diferenciar una medición.

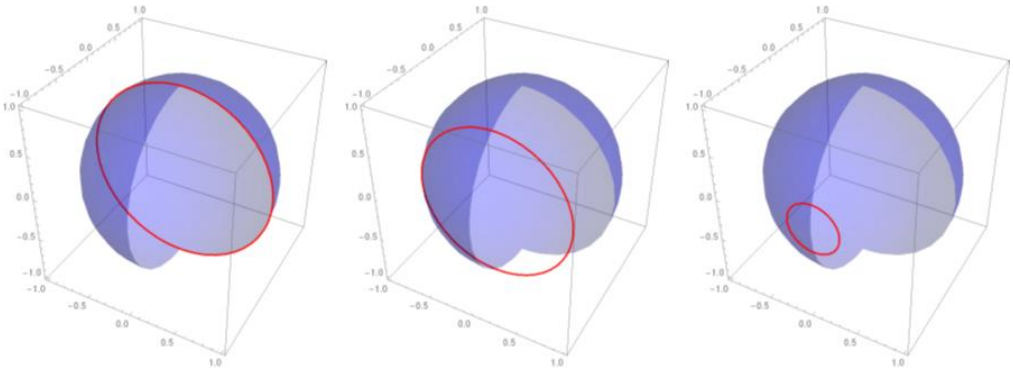


Figura 1.4 En rojo, valores que puede tomar una medición para una inclinación dada.

Cuando dos planos están unidos a través de una articulación rotacional simple, también llamadas bisagras, es posible determinar el ángulo presente entre ellos a partir de la inclinación de su superficie. Como el eje de rotación es el mismo para ambos planos, sus mediciones son mapeadas sobre el mismo círculo. Esto permite obtener los ángulos entre las dos superficies como si se tratara de hallar el ángulo entre dos vectores en un espacio 2D, a pesar de que se encuentran en un espacio 3D, lo cual permite determinar tanto la magnitud como el signo del ángulo. Para ello se cuenta con dos métodos: coordenadas esféricas y Kurata.

1.3.2. *Coordenadas esféricas*

Un sistema de coordenadas esféricas permite representar un punto P de un espacio 3D especificando 3 valores: la distancia al origen (r), el ángulo polar (θ) y el ángulo azimutal (ϕ), como se observa en la **Figura 1.5**.

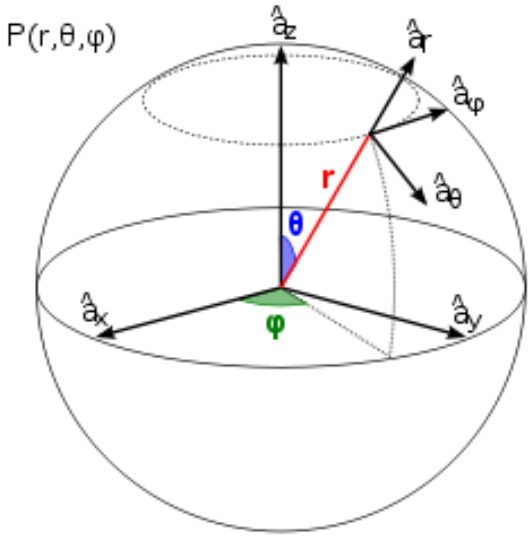


Figura 1.5 Relación de las coordenadas esféricas y las coordenadas cartesianas.

El acelerómetro registra las mediciones en coordenadas cartesianas. Si estás se transforman a coordenadas esféricas, asociando el eje de rotación al parámetro z de la función de transformación, de la forma: P(Az,Ay,Ax) → P(r, θ, ϕ) Se encuentra que el radio corresponde a la magnitud de la aceleración sensada, θ está asociado al ángulo de inclinación del eje de rotación y ϕ está asociado con el ángulo de rotación del plano. Una vez transformadas las mediciones de los acelerómetros a coordenadas esféricas, el ángulo entre los planos se obtiene mediante una diferencia entre ángulos ϕ, como se observa en la **Figura 1.6**.

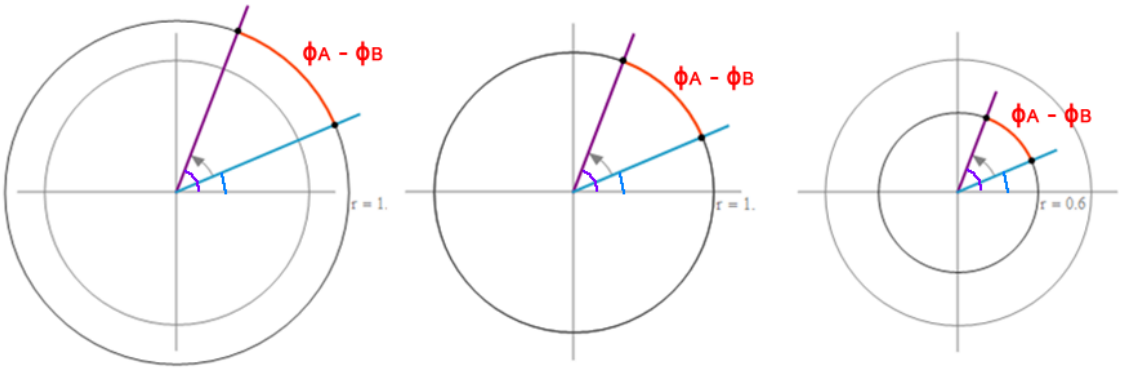


Figura 1.6 Obtención de ángulos con diferentes inclinaciones del eje de rotación.

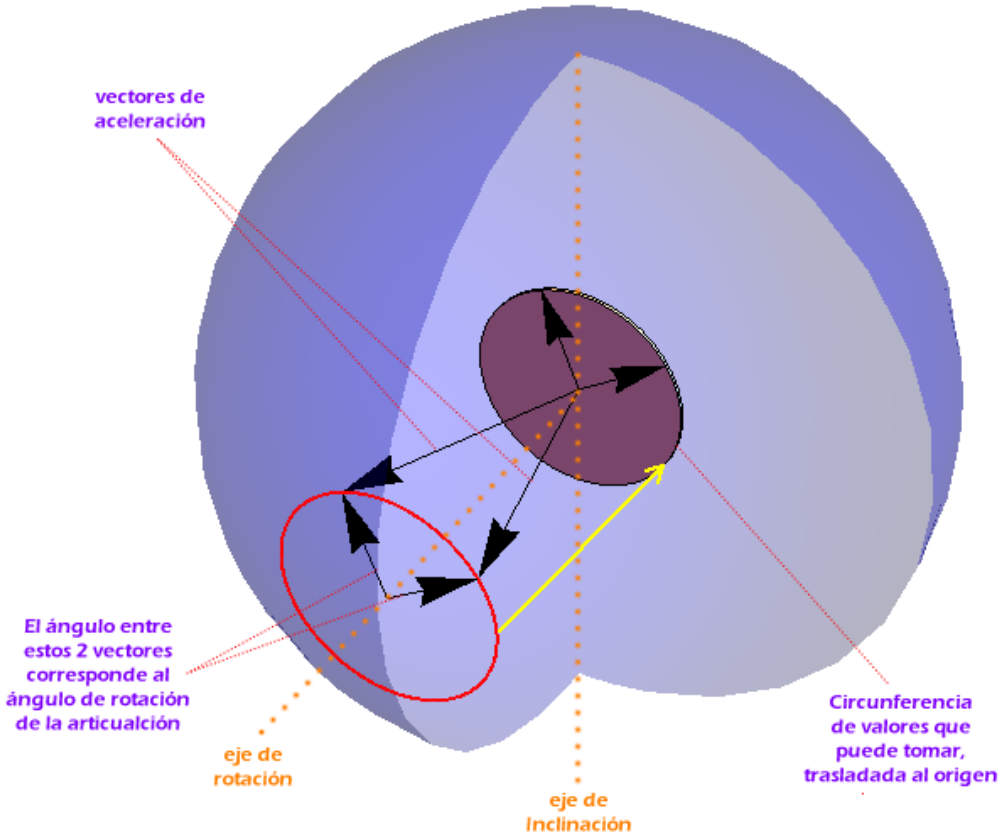
1.3.3. *Método de Kurata*

El método de Kurata se puede considerar una simplificación del método Producto-Vectorial / Producto-Punto (PvPp). El método PvPp calcula ángulos solo en un espacio 3D, debido al producto vectorial. Si al eje sobre el cual se ubica el eje de rotación, el cual no participa en el cálculo del ángulo, se le asigna el valor de cero, geométricamente se reduce el cálculo a un espacio 2D, ya que la circunferencia formada por los valores que puede tomar para una inclinación dada, es traslada al origen.

En la **Figura 1.7** el eje de rotación es el eje X, por lo que las componentes Ax y Bx deben tener el valor de 0 para trasladar la circunferencia al origen. Esto cancela 2 de los 3 términos subradicales del modelo PvPp, dejando un solo termino subradical. La operación de norma sobre un solo termino, es equivalente a la operación de valor absoluto, por lo que se pierde el signo del termino subradical. Este signo es importante, ya que indica cuál de los 2 vectores, A o B, es tomando como referencia al calcular el ángulo entre ambos.

$$\text{Ø} = \text{ArcTan}\left[\frac{\sqrt{\text{Abs}[-\text{Bx A1} + \text{Ax By}]^2 + \text{Abs}[\text{Bx Az} - \text{Ax Bz}]^2 + \text{Abs}[-\text{By Az} + \text{Ay Bz}]^2}}{\text{Ax Bx} + \text{Ay By} + \text{Az Bz}}\right]$$

Método PvPp: Angulo entre 2 vectores de aceleración



Modelo de Kurata: Angulo presente en la articulación

$$\text{Ø} = \text{ArcTan}\left[\frac{-\text{Az By} + \text{Ay Bz}}{\text{Ay By} + \text{Az Bz}}\right]$$

Figura 1.7. Obtención del módelo de kurata a partir del método PvPp

1.4. *Objetivos*

El objetivo general de este proyecto es desarrollar un guante electrónico basado en acelerómetros para determinar la postura de la mano a partir de la medición de los ángulos en las articulaciones. Este objetivo se acota mediante 4 objetivos específicos:

- ✓ Obtener datos fisiológicos de la inclinación de las falanges, palma y antebrazo a partir de las mediciones de la gravedad utilizando acelerómetros, con el fin de proveer en tiempo real información digital de los ángulos de la postura de la mano.
- ✓ Configurar una interfaz SPI, que permita centralizar los datos de la red de acelerómetros de forma concurrente y a la máxima tasa de muestreo.
- ✓ Construir un sistema de tecnología vestible, en el cual los componentes electrónicos son ubicados sobre un guante de acuerdo con los requerimientos de funcionalidad.
- ✓ Comprobar utilizando una representación visual, que el sistema permite determinar la postura de la mano para rotaciones sobre un plano vertical y considerando un grado de libertad por articulación.

1.5. *Mapa del proyecto*

En este proyecto se implementó un guante electrónico como plataforma de adquisición de datos, la cual permite obtener los vectores normales a la superficie dorsal de cada uno de los segmentos de la mano considerados y centralizar dichas mediciones en una FPGA. A partir de estas mediciones se calculan los ángulos presentes en las articulaciones, lo cual se implementó tanto de forma remota como de forma local (FPGA). Como mecanismo de verificación y validación, se desarrolló una representación visual de la mano, la cual permite visualizar de forma dinámica la postura de la mano sensada por el guante, lo que permite comparar de forma cualitativa la eficacia de la plataforma. En la **Figura 1.8** se presenta el mapa general de la solución implementada.

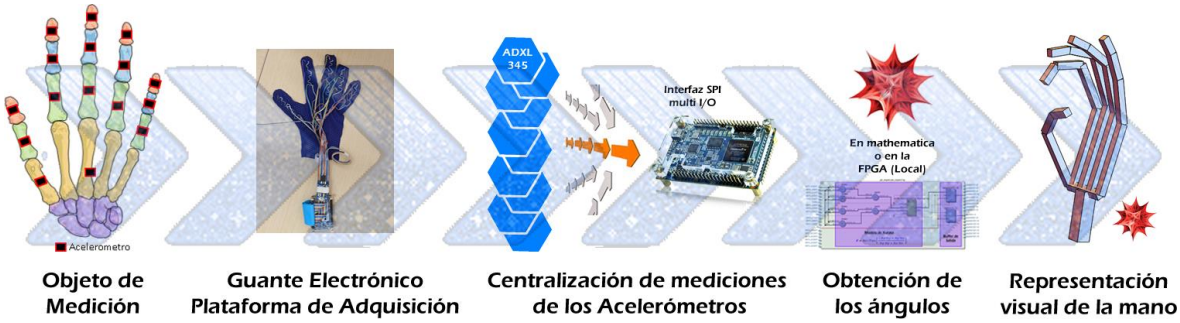


Figura 1.8 Mapa general del proyecto.

1.6. *Quartus II*

Las FPGAs son circuitos integrados compuestos de recursos digitales configurables por el diseñador. Existe una alta diversidad de arquitecturas de FPGAs, las cuales pueden estar compuestas de Flip-Flops, funciones lógicas, unidades aritméticas, memoria, interconexiones especializadas e I/O especializados. En este proyecto se utilizó una FPGA Cyclone IV de Altera.

Quartus II [8] es un conjunto de herramientas de automatización de diseño electrónico (EDA tools), desarrollada por Altera, una compañía dedicada a la manufactura de circuitos digitales reconfigurables como CPLDs y FPGAs. Quartus II permite al diseñador describir, sintetizar, depurar, verificar y programar los diseños; para lo cual utiliza herramientas como RTL Viewer, SignalTap, entre otras.

1.6.1. *RTL Viewer*

El RTL Viewer es una interfaz gráfica que permite examinar diagramas RTL (**Figura 1.9**) generados en el proceso de síntesis. Dicho diagrama RTL es un modelo estructural que el sintetizador ha generado a partir de la descripción realizada por el diseñador. Esta herramienta permite encontrar errores de descripción y garantizar que el hardware a implementar es acorde al diseño considerado por el diseñador.

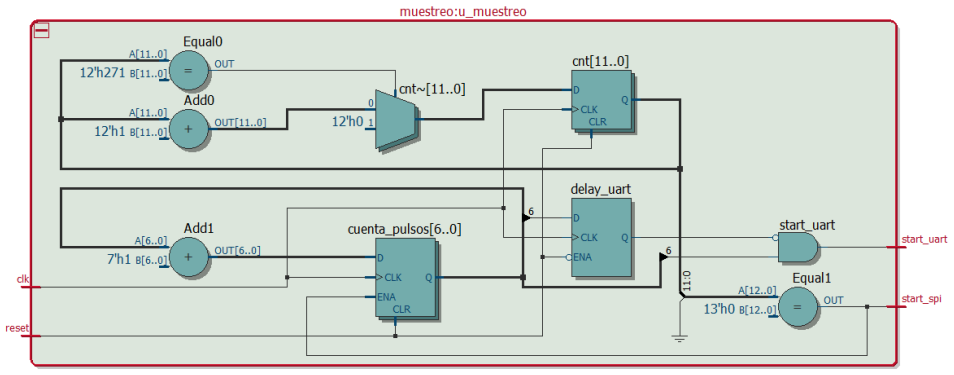


Figura 1.9 RTL Viewer

1.6.2. *SignalTap*

SignalTap es el analizador lógico de Quartus II. Permite analizar y verificar directamente sobre las FPGAs el funcionamiento de los diseños implementados. Analizando las señales se pueden identificar los errores, hasta verificar que el sistema cumple con los requerimientos. En la **Figura 1.10** se observa un muestreo de señales con SignalTap, utilizado en la verificación de la interfaz multi I/O SPI implementada.

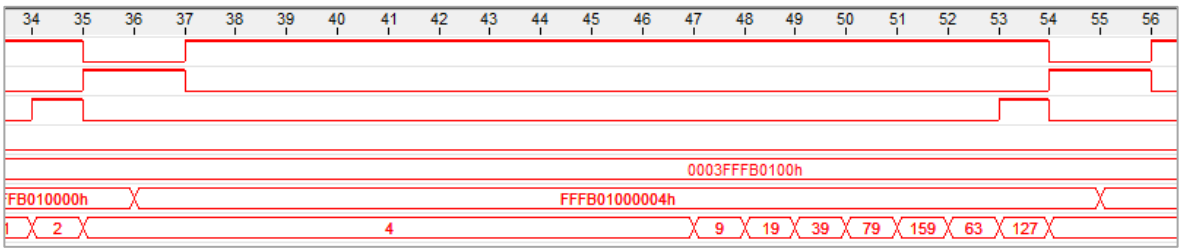


Figura 1.10 SignalTap

1.7. *Wofram Mathematica*

Mathematica [9] es un robusto y complejo sistema para computación técnica (No solo para matemáticas!) diseñado por Wolfram. Fue creado en 1988 y desde entonces ha mejorado continuamente. En cada nueva versión se incluyen nuevas funcionalidades y se siente como la plataforma evoluciona en función de los recursos tecnológicos actuales y las nuevas tendencias. Tradicionalmente, esta plataforma es considerada solo un sistema de algebra computacional, pero esto es solo la punta del iceberg. Mathematica también es un poderoso lenguaje de programación de propósito general. Las capacidades más útiles para el desarrollo de este proyecto son:

- ✓ La comunicación con dispositivos externos, por ejemplo: a través del puerto serial.
- ✓ La ejecución de procesos síncronos y periódicos.
- ✓ La visualización de la información de forma dinámica (monitoreo)

Actualmente, la universidad tiene un contrato de licencia que cobija a todos los PCs inventariados, además del derecho a una licencia para cada estudiante y funcionario para que instalen el software en sus PCs.

1.7.1. *Comunicación con dispositivos externos*

Hace algunos años, para poder comunicar Mathematica a través del puerto serial (puertos COM), era necesario instalar el paquete SerialIO o utilizar funciones de .NET, java o C, a través de un enlace con el kernel de Mathematica mediante el MathLink. Sin embargo, en la versión 10 de Mathematica (2014) se incluyó una serie de funciones que facilitan la comunicación a través del puerto serial, o con otros dispositivos, como la webcam. En este proyecto se utilizó estas funciones (**Figura 1.11**).

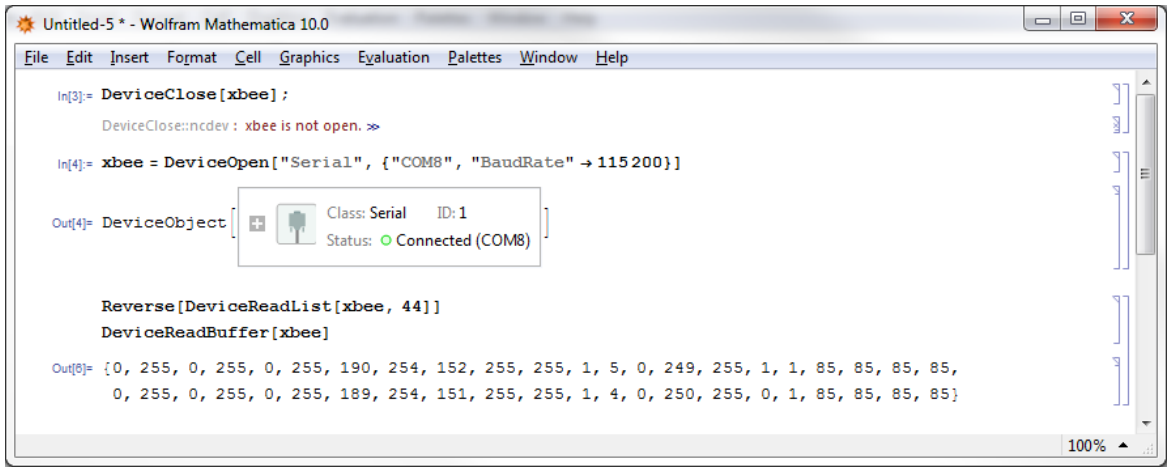


Figura 1.11 Adquisición de dato con Mathematica usando Devices Functions.

1.7.2. *Ejecución de procesos síncronos*

Mathematica permite la ejecución de tareas programadas (scheduled task), lo cual permite ejecutar periódicamente un determinado proceso. Estos procesos pueden ser interpretados como hilos, lo que mejora el rendimiento de ejecución. Sin embargo, es importante resaltar que las instrucciones a ser ejecutadas por una determinada tarea deben tener un tiempo de procesamiento menor al periodo de ejecución establecido para esta tarea. En la **Figura 1.12**, se observa como la función ImportString tarda más de 15ms en convertir un String en Integer16, mientras que la segunda instrucción lo hace tan rápido que no se puede determinar el tiempo de procesamiento.

```
In[12]:= ImportString["AC", "Integer16"] // Timing
         BinaryReadList[StringToStream["AC"], "Integer16"] // Timing

Out[12]= {0.015600, {17217}}

Out[13]= {0., {17217}}
```

Figura 1.12 Comparación del Tiempo de ejecución de la función ImportString con la solución propuesta.

1.7.3. *Visualización dinámica de la información*

La visualización de la información es un área importante en la interpretación de datos y es clave en las aplicaciones de monitoreo y análisis de información. Mathematica permite visualizar información de forma dinámica y en diferentes formatos. En la **Figura 1.13** se observa cómo la medición de un acelerómetro puede ser visualizada de forma dinámica y en diferentes formas, ya sea una lista de datos, una gráfica o una representación 3D.

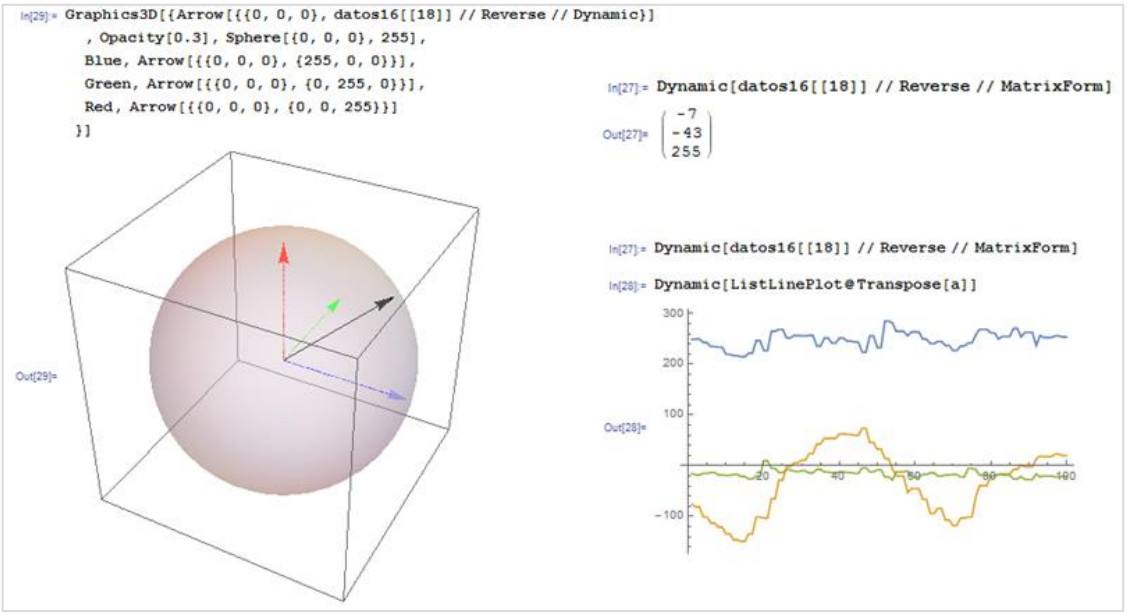


Figura 1.13 Diferentes formas de visualizar la información dinámicamente

1.8. *Referencias*

[1] "HAND POSTURE RECOGNITION USING ADABOOST WITH SIFT FOR HUMAN ROBOT INTERACTION." [Online]. Available: http://www.csie.ntu.edu.tw/~bobwang/Papers/wang_icar2007.pdf. [Accessed: 22-Sep-2015].

[2] A. F. da Silva, A. F. Goncalves, P. M. Mendes, and J. H. Correia, "FBG Sensing Glove for Monitoring Hand Posture," *IEEE Sens. J.*, vol. 11, no. 10, pp. 2442–2448, Oct. 2011.

[3] Dan Xu, Yen-Lun Chen, Xinyu Wu, Wei Feng, Huihuan Qian, and Yangsheng Xu, "A novel hand posture recognition system based on sparse representation using color and depth images," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 3765–3770.

[4] A. T. Velázquez, E. Merchán, and L. Hernández, "CARACTERIZACION CINEMATICA E IMPLEMENTACION DE UNA MANO ROBOTICA MULTIARTICULADA." 2008.

[5] A. E. Park, J. J. Fernandez, K. Schmedders, and M. S. Cohen, "The Fibonacci sequence: relationship to the human hand.," *J. Hand Surg. Am.*, vol. 28, no. 1, pp. 157–60, Jan. 2003.

[6] C. J. Fisher, "Using an Accelerometer for Inclination Sensing ," *AN-1057*, 2010. [Online]. Available: http://www.analog.com/media/en/technical-documentation/application-notes/AN-1057.pdf. [Accessed: 11-Jun-2015].

[7] M. Pedley, “Tilt Sensing Using a Three-Axis Accelerometer,” *AN3461*, 2013. [Online]. Available: http://cache-uat.freescale.com/files/sensors/doc/app_note/AN3461.pdf. [Accessed: 11-Jun-2015].

[8] “Quartus II Web Edition Software.” [Online]. Available: https://www.altera.com/products/design-software/fpga-design/quartus-ii/quartus-ii-web-edition.html. [Accessed: 11-Jun-2015].

[9] “Mathematica de Wolfram: El sistema definitivo para la computación técnica moderna.” [Online]. Available: http://www.wolfram.com/mathematica/?source=nav. [Accessed: 11-Jun-2015].

Capítulo 2
Centralización de los datos

Para determinar la postura de la mano se utilizaron diecisiete acelerómetros ADXL345, con el fin de obtener los vectores normales a la superficie dorsal de cada uno de los segmentos de la mano considerados. Las mediciones de aceleración (Ax, Ay y Az) son almacenadas en seis registros de ocho bits (Ax_H, Ax_L, Ay_H, AY_L, Az_H, Az_L) y se debe centralizar, en un solo dispositivo, las mediciones de los diecisiete acelerómetros. En este proyecto, estas mediciones se centralizan en una FPGA mediante una interfaz multi I/O SPI, configurada para diecisiete canales I/O (Un canal por acelerómetro). Dicha interfaz permite la centralización de estas mediciones de forma concurrente y a la máxima tasa de muestreo soportada por los acelerómetros, que es de 3200Hz.

Este capítulo inicia con una breve descripción de la interfaz SPI y de algunas interfaces basadas en ésta, propuestas por fabricantes de circuitos integrados. Luego se analiza la interfaz multi I/O SPI de 17 puertos y el sistema de corrección de offset, ambos diseñados e implementados. Finalmente, se presentan los resultados obtenidos con la implementación del sistema de centralización de datos.

2.1. Bus SPI

El bus SPI (Serial Peripheral Interface) es una especificación de interfaz de comunicación síncrona desarrollada por Motorola, y usado principalmente para la transferencia de datos entre circuitos integrados [1] [2]. La implementación del SPI no está regulada, por lo que existe una gran variedad de implementaciones, encontrando tramas de diferente tamaño y estructura. Sin embargo, hay elementos claramente establecidos. Un bus SPI típico está compuesto por cuatro puertos de conexión:

- ✓ CLK (Señal de reloj)
- ✓ MOSI (Master Output – Slave Input, línea de datos) o SDI (Serial Data Input)
- ✓ MISO (Master Input – Slave Output, línea de datos) o SDO (Serial Data Output)
- ✓ SS (Slave Select, selección del dispositivo esclavo)

Los primeros 3 puertos (CLK, SDI, SDO) son compartidos por todos los dispositivos, mientras que el puerto SS es usado para habilitar el dispositivo con el que se desea

comunicar. Si al bus se conectan tres dispositivos esclavos, el dispositivo maestro debe tener mínimo 3 puertos SS, uno por cada dispositivo esclavo, cómo se indica en la Figura 2.1. Esto significa que la cantidad de pines requeridos en el dispositivo maestro incrementa en al menos un pin por cada dispositivo esclavo conectado.

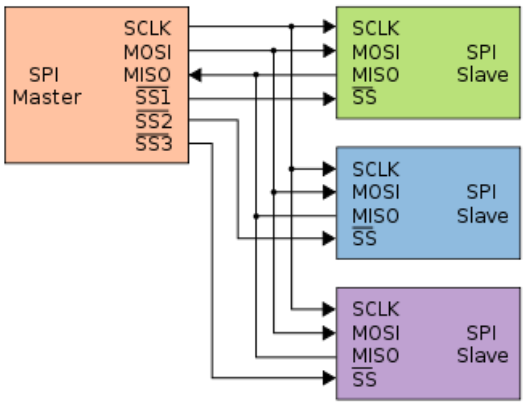


Figura 2.1 Bus SPI típico. Múltiples dispositivos esclavos.

Una operación de lectura o escritura (transacción) inicia con la selección del dispositivo esclavo con el que el dispositivo maestro se desea comunicar; esto se logra mediante la activación del puerto SS correspondiente. Luego de un tiempo, definido por el fabricante, se activa la señal de reloj con la cual se sincroniza la comunicación entre los 2 dispositivos y se inicia la transferencia de datos de forma serial. Dicha transferencia de datos está sincronizada con los flancos de la señal de reloj, generalmente con el flanco de subida. La transacción finaliza con la desactivación del respectivo puerto SS. Vale mencionar que el bus SPI, a diferencia del bus I2C, soporta el envío de múltiples bytes de datos por transacción.

2.2. Interfaces basadas en SPI

Existen muchos diseños basados en el bus SPI, de los cuales se analizan:

- ✓ SPI de 3 hilos.
- ✓ Múltiples interfaces SPI con conexión dedicada.
- ✓ Configuración Daisy-Chain.
- ✓ mSPI
- ✓ Multi I/O SPI

2.2.1. SPI de 3 hilos

En la configuración SPI de tres hilos, los puertos SDI y SDO, se concentran en un solo pin bidireccional llamado SDIO [3], como se observa en la Figura 2.2. Es principalmente útil

para aplicaciones donde el número de pines es muy limitado y en especial, con un solo dispositivo esclavo. Al incrementar el número de dispositivos esclavos también se incrementa el número de pines SS, por lo que empieza a perder valor la reducción de un pin de datos.

Figura 2.2 SPI 4 hilos (típico) vs SPI de 3 hilos.

2.2.2. Múltiples interfaces SPI con conexión dedicada

En esta configuración cada dispositivo esclavo se comunica con el dispositivo maestro mediante una interfaz SPI dedicada. Esta configuración es útil en aplicaciones: donde el ancho de banda del canal limita la transmisión de datos de múltiples dispositivos; donde es necesario incrementar el throughput, enviando los datos por múltiples interfaces; y donde es importante la independencia entre las diferentes interfaces SPI

El tiempo necesario para comunicarse con cada uno de los dispositivos conectados no se ve afectado por la cantidad de dispositivos, ya que la comunicación con cada dispositivo se lleva a cabo de forma paralela. Entre las desventajas se encuentra el incremento en los requerimientos de hardware, debido a la cantidad de pines utilizados y al uso de múltiples interfaces SPI en el dispositivo maestro. La alta configurabilidad de las FGPAs, tanto a nivel de lógica como de pines, hace posible la implementación de este tipo de interfaces (Ver Figura 2.3).

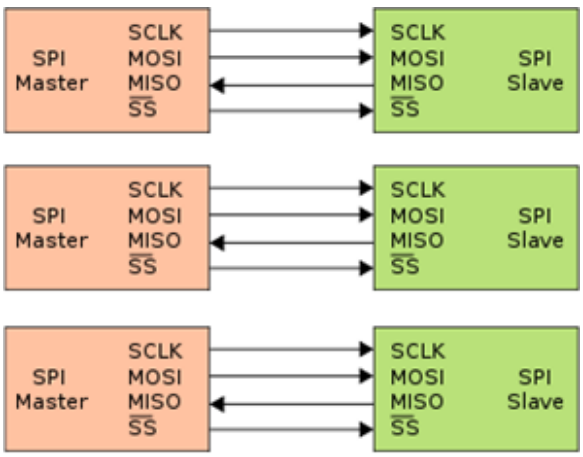


Figura 2.3 Múltiples interfaces SPI con conexión dedicada.

2.2.3. Configuración Daisy-Chain

Una interfaz SPI cuenta con 2 registros de desplazamiento: uno para datos de entrada y otro para datos de salida. El registro de desplazamiento de datos de entrada desempeña el papel de un conversor serial a paralelo (SDI -> Byte) y el registro de desplazamiento de datos de salida se comporta como un conversor paralelo a serial (Byte -> SDO). En la configuración Daisy-Chain se utiliza un solo registro de desplazamiento para los datos de entrada y de salida (SDI -> Byte -> SDO).

Dicha configuración posibilita la conexión de los dispositivos en cascada, como se observa en la Figura 2.4, lo que evita la posibilidad de cortos debidos a problemas de control de acceso al medio. Como es necesario que todos los dispositivos estén habilitados durante la transmisión de datos, se requiere de un solo puerto SS para la habilitación de todos los dispositivos esclavos, es decir, el dispositivo maestro no se ve afectado a nivel de pines por el número de dispositivos esclavos conectados. El ADXL345, dispositivo SPI esclavo utilizado en este proyecto, no soporta la configuración Daisy-Chain.

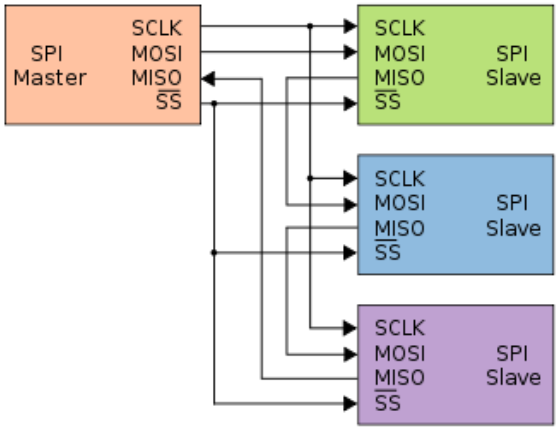


Figura 2.4 Configuración Daisy-Chained.

2.2.4. mSPI (mini-SPI)

En el bus mSPI se agrega al inicio de la trama un campo adicional llamado “dirección de esclavo” el cual permite identificar al dispositivo a quien va dirigida la trama, por lo tanto solo un dispositivo esclavo lee o escribe datos en el bus y no es necesario el uso de múltiples pines SS para el control de acceso al medio. Esta configuración elimina el problema de incremento de los pines en función del número de dispositivos esclavos (Figura 2.5) al igual que la configuración Daisy-Chain, pero reduce el throughput debido al incremento de longitud de la trama.

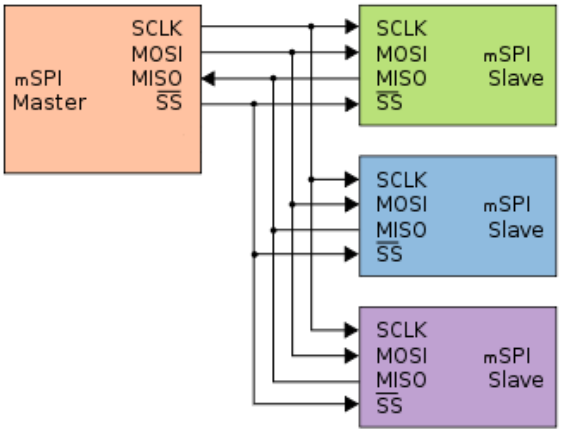


Figura 2.5 Configuración mini-SPI.

2.2.5. Multi I/O SPI

En esta configuración se utiliza un solo puerto SS y CLK para todos los puertos I/O, lo que implica que los puertos I/O transmiten datos simultáneamente. Su implementación es similar al SPI de 3 hilos, pero se diferencia en que presenta múltiples puertos de datos I/O. Mientras más puertos I/O, mayor es el flujo de datos transferidos, lo que permite incrementar el throughput de la interfaz. Un caso especial de esta configuración es el SQI (Serial Quad I/O) [5] [6], el cual se utiliza en las SD cards para incrementar en 4 veces el throughput (Figura 2.6).

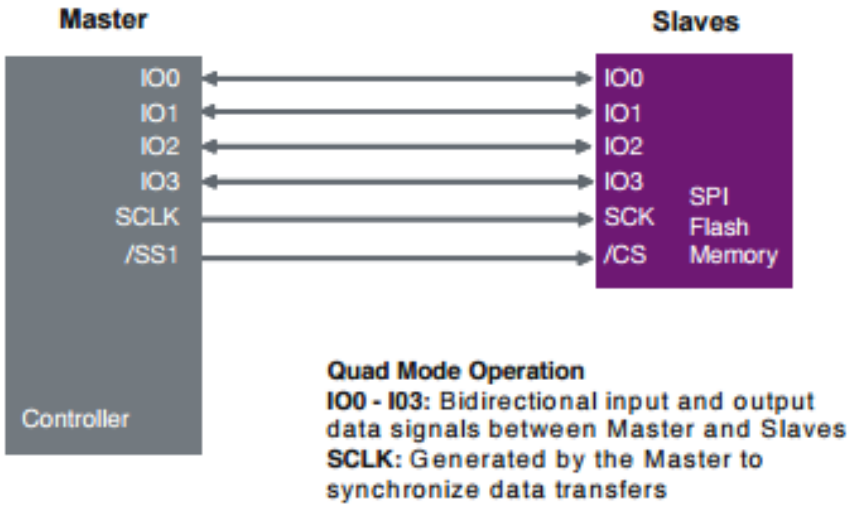


Figura 2.6 Multi I/O SPI.

2.3. Interfaz multi I/O SPI para la centralización de datos de 17 acelerómetros ADXL345.

Se diseñó e implementó una interfaz multi I/O SPI para la centralización de las mediciones de los 17 acelerómetros utilizados para determinar la postura de la mano. El número de canales I/O de dicha interfaz es parametrizable, configurado en este proyecto para 17 canales I/O. Esta interfaz está compuesta por una unidad de control, un datapath y un registro de almacenamiento paralelo, como se observa en la Figura 2.7.

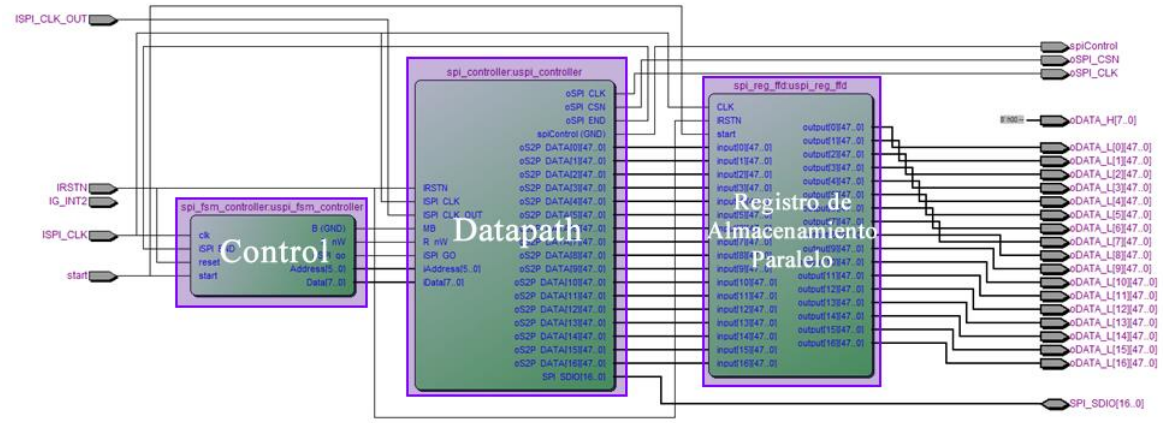


Figura 2.7 Unidad de control de la interfaz SPI.

2.3.1. Unidad de control

La unidad de control es la encargada de gestionar la secuencia de registros que se deben consultar en el acelerómetro ADXL345, tanto en los procesos de configuración (Escritura) como de centralización de datos (Lectura). La unidad de control está compuesta por un contador, una lógica de salida y una máquina de estados finita (FSM).

El valor del contador define qué registro del acelerómetro se va a configurar o leer durante la presente transacción. La constante Setup_End indica el número de instrucciones de configuración y la constante Step_End la cantidad total de instrucciones; La diferencia entre Step_End y Setup_End corresponde al número de instrucciones de solicitud de datos (adquisición). Los primeros valores del contador, que van desde 0 hasta Setup_End - 1, están asociados a instrucciones de configuración de los acelerómetros, mientras que los valores del contador entre Setup_End y Step_End están asociados a solicitudes de lectura de datos (Ver Figura 2.8).

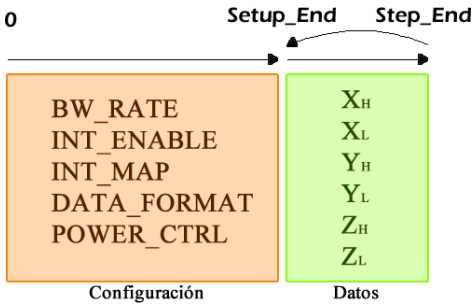


Figura 2.8 Flujo de configuración y consulta de los registros del acelerómetro.

La lógica de salida corresponde a los datos que se deben entregar al datapath para ser transmitidos por la interfaz SPI: indicar si la operación es de lectura o escritura (R/nW, 1 bit), indicar si se van a leer o escribir múltiples registros en el ADXL345 (MB, 1 bit), la dirección del registro que se va a leer o escribir (Address, 6 bits) y los datos que se van a enviar (Data, 8 bits), los cuales solo tienen importancia en operaciones de escritura.

La FSM implementada se presenta en Figura 2.9. Dicha FSM está compuesta por 4 estados y 3 estímulos de transición. El proceso de adquisición (S1) se inicia mediante un estímulo en el pin Start. Dicho estímulo está asociado al tiempo de muestreo, es decir, se presenta cada vez que se requiere consultar los registros de aceleración de los acelerómetros, por ejemplo, cada 10ms para una tasa de muestreo de 100Hz. La FSM permanece en el estado S1 mientras el datapath se encuentre transmitiendo datos. Cuando la transmisión de datos termine, el datapath informa dicho evento a la unidad de control mediante el flag SPI_END, pasando al estado de fin de transmisión (S2). En este estado incrementa en uno el contador (cnt) y compara su valor con la constante Step_End; si cnt es menor que Step_end, significa que no sé han consultado todos los registros y debe regresar a S1 para llevar a cabo la adquisición de un nuevo dato; si cnt es igual a Step_End, significa que ya se consultaron todos los datos y se debe regresar al estado inicial (S0) y esperar hasta el próximo ciclo de muestreo. Antes de pasar al estado S0, se asigna al contador el valor Setup_End, con el fin de que el proceso de configuración del acelerómetro solo se lleve a cabo en el primer ciclo de muestreo. El estado S3 solo se usa para generar una pausa de un ciclo de reloj entre cada operación de consulta de datos.

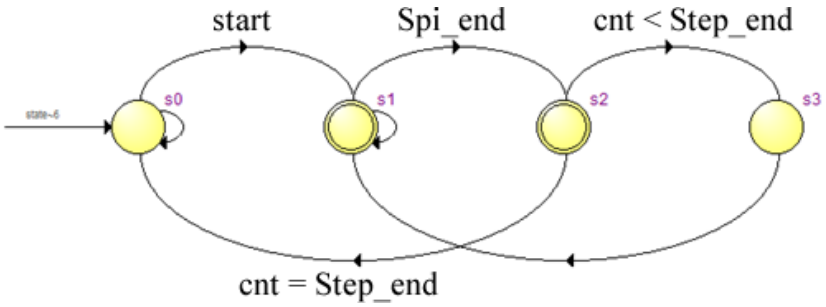


Figura 2.9 FSM del controlador de la interfaz SPI.

2.3.2. Datapath de interfaz SPI

La comunicación SPI entre el acelerómetro ADXL345 y la FPGA se lleva a cabo mediante tramas que cumplan con el formato presentado en la figura Figura 2.10. A partir del análisis RTL de un Proyecto de demostración de la DE0-nano para la comunicación con el ADXL345, codificado en Verilog, se codificó en VHDL un datapath equivalente. Dicho diseño del datapath no soporta la comunicación SPI de múltiples bytes. A continuación se analiza el funcionamiento del datapath.

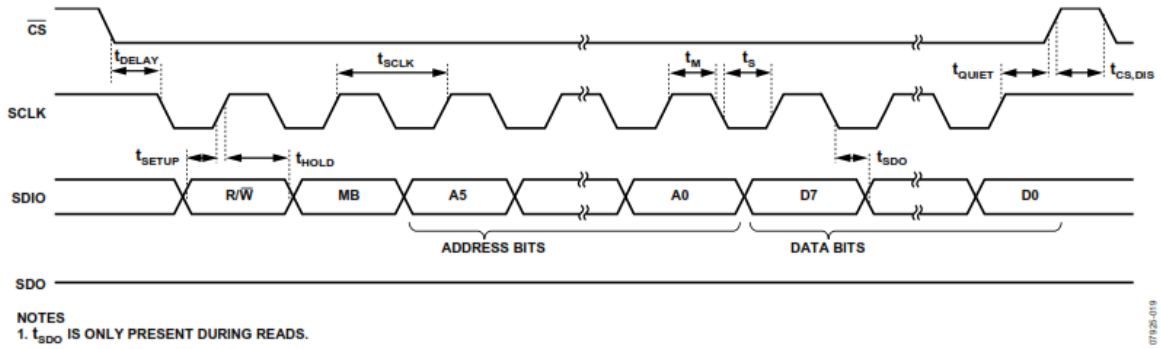


Figura 2.10 Formato de trama para comunicación SPI de 3 hilos de los ADXL345.

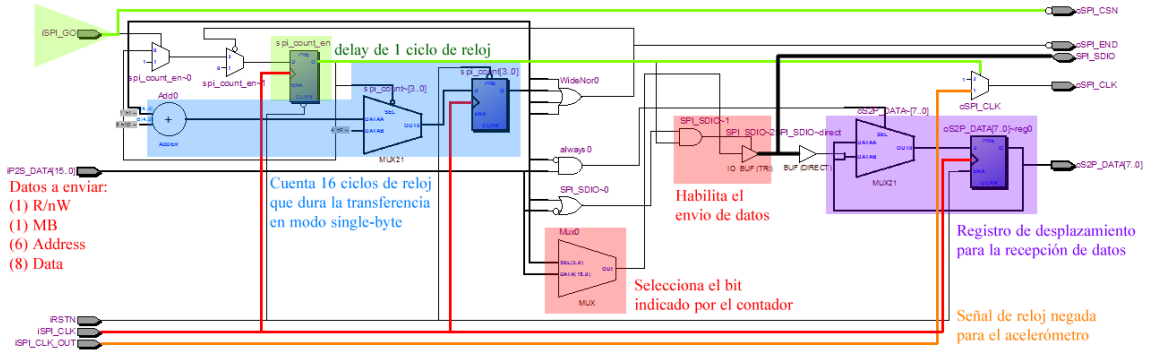


Figura 2.11 Datapath de la unidad de control de la interfaz SPI.

La transmisión inicia cuando la unidad de control activa el pin SPI_GO, indicado con un triángulo verde en la Figura 2.11; dicha señal está asociada al pin SS del esclavo, por lo que debe permanecer activa durante todo el proceso de transmisión. Al iniciar la transmisión, se inicia un contador con el valor quince. El valor del contador se usa para seleccionar el bit que debe ser transmitido en cada ciclo de reloj, controlando el flujo de datos de un multiplexor de dieciséis entradas de un bit.

Un buffer triestado habilita la transmisión del bit seleccionado, solo cuando el dispositivo maestro debe estar transmitiendo. Cuando el dispositivo maestro está recibiendo, como sucede en los últimos ocho bits de una operación de lectura, la salida del buffer triestado debe estar en alta impedancia, evitando la generación de cortos sobre el puerto SDIO, el

cual es bidireccional. Los datos recibidos de forma serial por el pin SDIO (lectura) son almacenados en un registro de desplazamiento de 8 bits, lo que permite acceder a estos de forma paralela (1 byte).

2.3.3. Registro de desplazamiento

El datapath solo registra el último byte adquirido, pero los datos de aceleración están almacenados mediante 6 bytes. Se implementó un registro de desplazamiento para datos de 1 byte, con capacidad de 6 bytes, de tal manera que "al final" del ciclo de lectura, los 6 bytes almacenados en dicho registro corresponden a las mediciones de los acelerómetros tomadas en un mismo ciclo de muestreo.

2.3.4. Datapath de Interfaz Multi I/O SPI

El datapath de la interfaz SPI (analizado en 1.3.2) se encarga del proceso de comunicación de un solo canal SDIO y el registro de desplazamiento de pre-almacenar los datos muestreados por el respectivo canal SDIO. Estas tareas se deben replicar 17 veces, con el fin de dedicar un puerto SDIO a cada acelerómetro (Ver Figura 2.12). Debido a que los datos enviados por la FPGA a cada uno de los dispositivos esclavos son los mismos, se puede usar la misma fuente de datos para todos los módulos, es decir, se puede utilizar la misma unidad de control para cada uno de los datapath SPI contenidos en la interfaz multi I/O SPI.

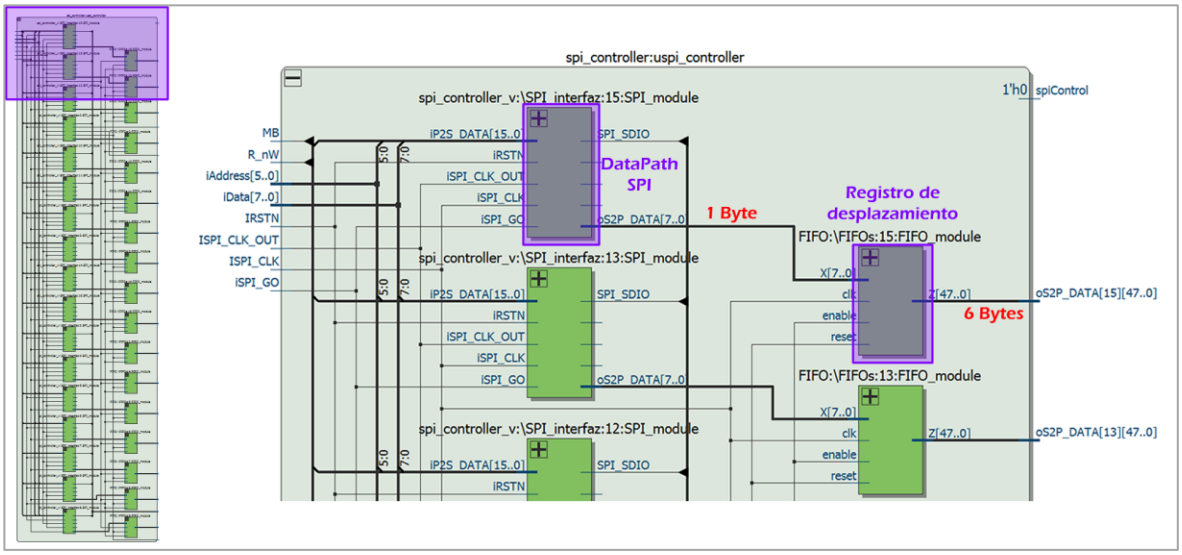


Figura 2.12 Datapath de la unidad de control de la interfaz multi I/O SPI.

2.3.5. Registro de almacenamiento paralelo

El objetivo de este registro de desplazamiento paralelo es registrar datos coherentes para las etapas de procesamiento posteriores. Con datos coherentes se hace referencia a que

su contenido siempre sea del tipo {XH-XL-YH-YL-ZH-ZL}. En el registro de desplazamiento (2.3.3.) solo se alcanza este estado de coherencia después de adquirir los 6 bytes de datos. Por lo tanto, se implementó una lógica que permite actualizar los datos de este registro de almacenamiento paralelo luego de que el registro de desplazamiento adquiera los 6 bytes de datos asociados a un ciclo de muestreo.

2.4. Corrección de Offset

Los acelerómetros presentan un desplazamiento en la medición debido a la temperatura [3], como se presenta en la Figura 2.13, donde se observan errores de offset en el eje Z entre 950 y 1100 a temperatura ambiente. Este offset se corrige restando el valor adecuado a los datos medidos, como se indica en la Figura 2.14. En este proyecto se implementaron dos estrategias para corregir el offset. La sugerida por el fabricante y la desarrollada a través de métodos experimentales.

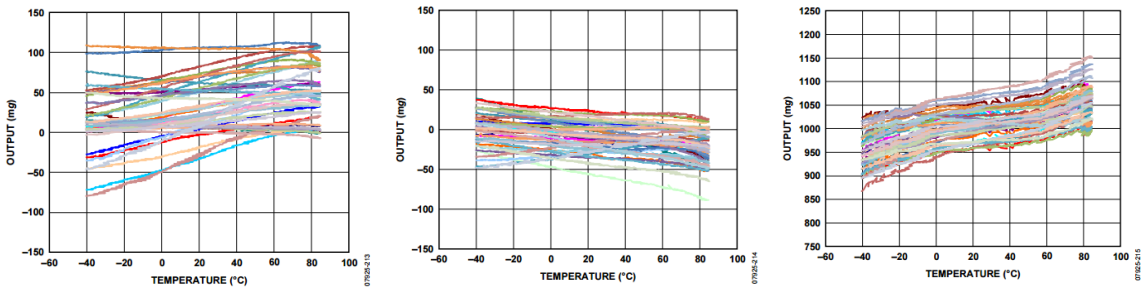


Figure 13. X-Axis Zero g Offset vs. Temperature—45 Parts Soldered to PCB, V_S = 2.5 V

Figure 14. Y-Axis Zero g Offset vs. Temperature—45 Parts Soldered to PCB, V_S = 2.5 V

Figure 15. Z-Axis One g Offset vs. Temperature—45 Parts Soldered to PCB, V_S = 2.5 V

Figura 2.13 Variación del offset con la temperatura.

$$Ax = Medición\_Ax - Offset\_Ax$$

$$Ay = Medición\_Ay - Offset\_Ay$$

$$Az = Medición\_Az - Offset\_Az$$

Figura 2.14. Corrección del offset.

2.4.1. Implementación de sistema de corrección de offset. Sugerencia del fabricante

El proceso sugerido por el fabricante para la corrección del offset es colocar el acelerómetro lo más horizontal posible y ajustar el offset de tal manera que la medición sea {Ax,Ay,Az} = {0,0,255}, lo cual corresponde a una medición teóricamente ideal. Esto equivale a asignar un offset igual a { -Ax , -Ay , -Az+255 }. Se implementó un circuito que permite al usuario ajustar el offset mediante comandos enviados por la interfaz UART (Figura 2.15), utilizando el algoritmo anteriormente descrito y las mediciones actuales de los acelerómetros.

- ✓ El comando “0X” ajusta el offset de todos los acelerómetros
- ✓ El comando “1X” ajusta el offset de los acelerómetros ubicados sobre el dedo meñique.
- ✓ El comando “2X” ajusta el offset de los acelerómetros ubicados sobre el dedo anular.
- ✓ El comando “3X” ajusta el offset de los acelerómetros ubicados sobre el dedo corazón.
- ✓ El comando “4X” ajusta el offset de los acelerómetros ubicados sobre el dedo índice.
- ✓ El comando “5X” ajusta el offset de los acelerómetros ubicados sobre el dedo pulgar.
- ✓ El comando “6X” ajusta el offset de los acelerómetros ubicados en la palma y el antebrazo.

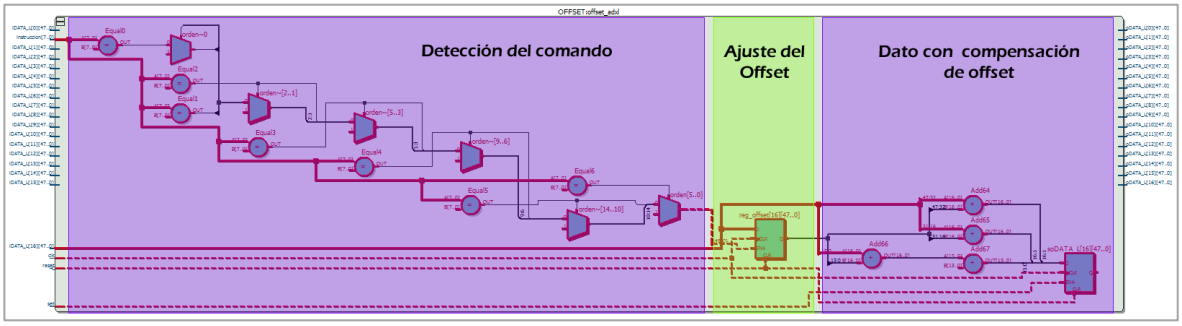


Figura 2.15 Hardware que permite ajustar el valor para compensar el offset.

Sin embargo, este procedimiento no permite corregir adecuadamente el offset en todos los dispositivos por la siguiente razón: En el proceso de calibración se observa que la gravedad generaba un efecto variable en cada uno de los acelerómetros, generando mediciones estáticas con magnitudes entre 240 y 300, por lo que el método anterior no permite corregir el problema del offset.

2.4.2. Implementación de sistema de corrección de offset. Procedimiento experimental.

Para corregir el offset se implementó un proceso más preciso. Se registraron las mediciones con el eje Z hacia arriba y hacia abajo (Sin problemas de offset, A_X y A_Y deben medir un valor de 0) y con el eje X hacia arriba y hacia abajo (A_Y y A_Z = 0). A partir de la comparación de estas mediciones se calculó un valor para el offset, el cual se configuró por defecto en el sistema. El resultado fue realmente bueno, permitiendo al sistema obtener la medición de la inclinación de forma correcta, tarea fundamental para la medición de ángulos a partir de las mediciones de inclinación. Vale aclarar que el sistema de corrección de offset basado en la sugerencia del fabricante también está habilitado, lo que permite re-ajustar el offset mediante los comandos mencionados previamente. Para volver a los valores por defecto, basta con des-energizar la FPGA.

2.5. Resultados

La interfaz multi I/O SPI permite la centralización de datos de los 17 acelerómetros de forma concurrente. Esto permite llevar a cabo el proceso de centralización de datos a todas las tasas de muestreo soportadas por el acelerómetro ADXL345, donde la tasa de muestreo más alta es de 3200Hz. En la Figura 2.16 se presenta la adquisición de los datos de 17 acelerómetros a un muestreo de 3200Hz, con la interfaz Multi I/O SPI implementada. Se puede observar que por cada puerto I/O se pueden transmitir los datos de máximo 5 acelerómetros, por lo que sería necesario el uso de al menos cuatro interfaces SPI típicas para llevar a cabo el proceso de centralización a la máxima tasa de muestreo.

Con la interfaz multi I/O SPI, se utiliza la misma unidad de control en la adquisición de datos para un solo acelerómetro como para diecisiete acelerómetros ADXL345, lo que simplifica la escalabilidad del sistema de centralización de datos.

El uso de recursos es principalmente de flip-flops. Para una configuración de 17 canales I/O se utilizaron 60 Elementos lógicos (LEs) y 1783 flip-flops. El dispositivo tiene 22320 LEs y flip-flops, por lo que el uso de LEs es menor al 1% y el de flip-flops es menor al 8%.

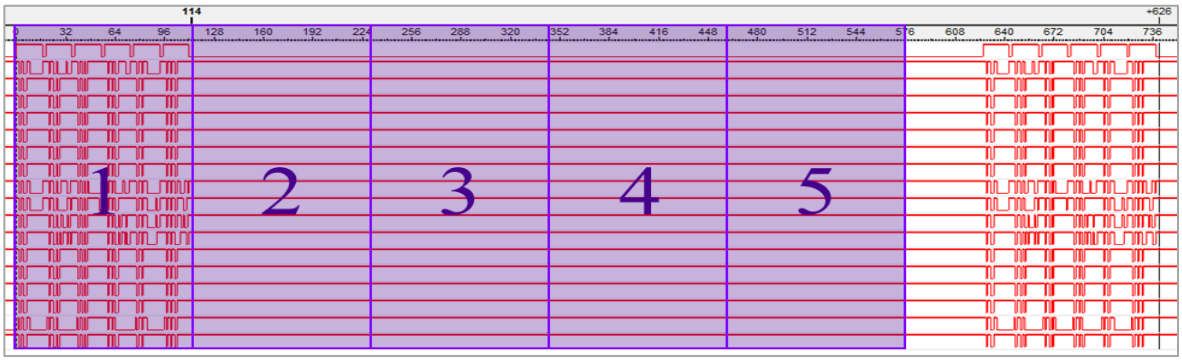


Figura 2.16. Muestreo a 3200Hz usando la interfaz Multi I/O SPI implementada.

2.6. Referencias

[1] Oudjida, A.K.; Berrandjia, M.L.; Tiar, R.; Liacha, A.; Tahraoui, K., "FPGA implementation of I^2C & SPI protocols: A comparative study," *Electronics, Circuits, and Systems, 2009. ICECS 2009. 16th IEEE International Conference on* , vol., no., pp.507,510, 13-16 Dec. 2009

[2] Blessington, T.P.; Murthy, B.B.; Ganesh, G.V.; Prasad, T.S.R., "Optimal implementation of UART-SPI Interface in SoC," *Devices, Circuits and Systems*

(ICDCS), 2012 International Conference on , vol., no., pp.673,677, 15-16 March 2012

[3] "ADXL345 Datasheet." [Online]. Available: http://www.analog.com/media/en/technical-documentation/data-sheets/ADXL345.pdf. [Accessed: 11-Jun-2015].

[4] Xiaofei Wang; Hong Zhang; Long Zhang; Jianrong Zhang; Yue Hao, "A daisy-chain SPI interface in a battery voltage monitoring IC for electric vehicles," *Solid-State and Integrated Circuit Technology (ICSICT), 2014 12th IEEE International Conference on* , vol., no., pp.1,3, 28-31 Oct. 2014

[5] Xiaohu Wang; Zhaoming Huang, "Design and implementation of high speed QSPI memory controller," *Electronics Information and Emergency Communication (ICEIEC), 2013 IEEE 4th International Conference on* , vol., no., pp.82,85, 15-17 Nov. 2013

[6] Serial Peripheral Interface (SPI) Flash Memory Backgrounder, Spansion.

Capítulo 3
Calculo de los ángulos de las articulaciones

En el capítulo anterior se centralizó los datos adquiridos por los acelerómetros, correspondientes a la inclinación de la superficie dorsal de todos los segmentos de la mano. En este capítulo se realiza el cálculo de los ángulos presentes en las articulaciones a partir de los datos centralizados, y se utilizan 2 métodos: El primer método está basado en una transformación a coordenadas esféricas (Capitulo 1, Sección 1.3.2.), mientras que el segundo método se basa en el modelo propuesto por Kurata (Capitulo 1, Sección 1.3.3.)

El cálculo de los ángulos se llevó a cabo de forma remota (PC) y de forma local (FPGA). El cálculo remoto se implementó sobre la herramienta computacional Mathematica, mientras que el cálculo local se implementó en la tarjeta de desarrollo DE0-nano.

Para el cálculo de los ángulos en la FPGA se implementó una arquitectura de procesamiento concurrente basada en el modelo propuesto por Kurata, el cual involucra el uso de una función arcotangente. Se propuso un nuevo algoritmo para la implementación de la función arco-tangente, el cual se basa en aspectos simétricos relacionados con la distancia angular a la diagonal de las funciones tangente y cotangente. Se diseñó y se implementó en hardware un componente, basado en el algoritmo propuesto, que calcula la función arco-tangente y tiene una resolución de 1°, lo cual es suficiente para esta aplicación.

Por último, se implementó una representación visual 3D de la mano en Mathematica para validar el funcionamiento del sistema cualitativamente.

3.1. Caso 1. Cálculo de los ángulos con Mathematica.

Para calcular los ángulos en Mathematica se utilizó tanto el método basado en coordenadas esféricas como el método basado en el modelo propuesto por Kurata. Para poder calcular estos ángulos es necesario que las mediciones de los acelerómetros, las cuales están centralizadas en la FPGA, estén disponibles en Mathematica. Para transferir las mediciones de los acelerómetros desde la FGPA hasta Mathematica se utilizó una interfaz de comunicación UART.

3.1.1. Trama transmitida

Los datos son enviados a Mathematica mediante una secuencia de bytes (trama) que debe ser interpretada correctamente, reconociendo a qué corresponde cada byte. Para establecer esta sincronización se usó una cabecera de 6 bytes, la cual permite identificar donde inicia la trama, y es prácticamente imposible que ésta se presente entre los datos. En la Figura 3.1 se presenta la trama utilizada para transferir las mediciones de los acelerómetros, la cual tiene una longitud de 108 bytes. La cabecera de 6 bytes utilizada es {170, 85, 85, 85, 85, 170}; los 102 bytes restantes corresponden a las mediciones tri-axiales de los 17 acelerómetros (6 bytes cada una).

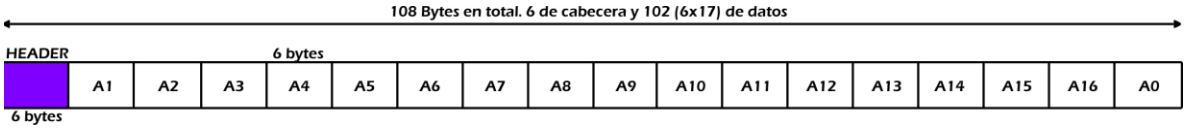


Figura 3.1 Trama utilizada para transferir las mediciones de inclinación

3.1.2. Adquisición y organización de los datos

El primer paso para interpretar los datos de la trama es identificar la cabecera, para lo cual se implementaron 2 métodos: el primer método consiste en leer el buffer de la interfaz UART byte por byte hasta detectar la secuencia de la cabecera. Este método es bastante ineficiente, con tiempos de procesamiento que oscilan entre 150ms y 400ms, lo cual solo permite ratas de muestreo menores a 2Hz. Este rendimiento tan bajo se debe a la gran cantidad de operaciones de acceso al puerto serial (UART).

Como solución se implementó el segundo método, el cual es un robusto y eficiente algoritmo para el proceso de adquisición, el cual no usa bucles, presenta pocos accesos al puerto serial y su tiempo de procesamiento es tan bajo, que no puede ser medido por la funcion Timing de Mathematica.

En la Figura 3.2 se presenta el algoritmo implementado (segundo método). Primero, se trasladan los datos del buffer del puerto serial a la variable bufferAux; esto reduce el número de accesos a dicha interfaz y evita la pérdida de datos por desbordamientos del buffer del puerto serial. Luego, mediante el uso de una función eficiente para la búsqueda de secuencias, se busca en la variable bufferAux (array) las posiciones donde se presenta la secuencia de la cabecera, lo que permite identificar donde inicia cada trama. Por último, si hay 108 bytes en el buffer a partir de la posición donde inicia la trama, es decir, si la trama está completa, entonces son movidos los 108 bytes a la variable trama.

```
seqpos[a_List, seq_List] := ReplaceList[a, {x___, Sequence @@ seq, ___} :> 1 + Length[{x}]];
xbee = DeviceOpen["Serial", {"COM7", "BaudRate" → 115200}];

RunScheduledTask[
  bufferAux = AppendTo[bufferAux, DeviceReadBuffer[xbee]] // Flatten;
  i = seqpos[bufferAux, {170, 85, 85, 85, 85, 170}];
  If[Length[i] == 0, i = {1}];
  bufferAux = Drop[bufferAux, i[[1]] - 1];
  If[Length[bufferAux] ≥ 108,
   trama = Take[bufferAux, 108]; bufferAux = Drop[bufferAux, 108];
  ]
  , 0.04
];
```

Figura 3.2. Algoritmo utilizado para la adquisición de las tramas con mediciones de inclinación.

Los datos adquiridos (stream de bytes) se deben interpretar como enteros de 16 bits (stream de Integer16). La instrucción ImportString es la función tradicionalmente utilizada para interpretar los datos como Integer16, pero presenta tiempos de procesamiento mayores a 15ms (**Figura 1.12**), por lo que solo puede usarse con tasas de muestreo menores a 50Hz. Se implementó un eficiente algoritmo para interpretar el stream de bytes (Ver Figura 3.3). Primero, la variable trama es interpretada como un stream de bits (datos en binario, sin formato) y luego, el stream de bits es interpretado como una secuencia de enteros de 16 bits y es almacenado en la variable datos16in.

```
datoR = trama // FromCharacterCode;
dato16in = BinaryReadList[StringToStream[datoR], "Integer16"];
datos16 = Partition[dato16in, 3];
datos16[[18]] = {datos16[[18, 1]], datos16[[18, 3]], -datos16[[18, 2]]};
aux = Insert[datos16, datos16[[-2]], {{-2}, {-2}, {-2}, {-2}}];
```

Figura 3.3 Interpretación de los datos como Integer16 y organización estratégica de los datos.

La lista de datos de la variable datos16in se dividió en filas de 3 elementos. Cada fila está asociada a los datos de un acelerómetro en particular, y corresponden a las mediciones {Az, Ay, Ax}. Las filas 2 al 6 están asociadas a las falanges distales, 7 al 11 a las falanges mediales, 12 al 16 a las falanges proximales, 17 a la palma de la mano y 18 al antebrazo. Se replicó cuatro veces la fila de datos asociada al acelerómetro de la palma (17); con esto se logró que las mediciones de los dos acelerómetros involucrados en el cálculo de un ángulo en particular, estén a una distancia de 5 filas para todos los casos, como se observa en la **Tabla 3.1**.

La organización matricial de los datos en filas y la replicación de las mediciones del acelerómetro asociado a la palma de la mano, permiten que los algoritmos utilizados para obtener los ángulos, se puedan implementar utilizando instrucciones basadas en listas. Mathematica, al igual que MATLAB, está optimizado para operaciones basadas en listas.

Se implementaron 2 algoritmos diferentes para obtener los ángulos: utilizando coordenadas esféricas y utilizando el modelo propuesto por Kurata.

	Distal	Medial	Proximal	Palma	Antebrazo
Meñique	2	7	12	17	22 (18)
Anular	3	8	13	18 (17)	
Corazón	4	9	14	19 (17)	
Indice	5	10	15	20 (17)	
Pulgar	6	11	16	21 (17)	

Tabla 3.1. Asociación de los segmentos de la mano con las mediciones de cada fila, luego de replicar 4 veces la fila 17.

3.1.3. Cálculo de ángulos mediante coordenadas esféricas

Cada fila tiene la estructura de datos (Az, Ay, Ax); su representación en coordenadas esféricas (r, θ, ϕ) corresponde a la magnitud de aceleración sensada (r), al ángulo de inclinación del eje de rotación (θ) y al ángulo de rotación (ϕ). En la Figura 3.4 se presenta el algoritmo utilizado. Primero se transforma las mediciones de aceleración, obtenidas en coordenadas cartesianas, a coordenadas esféricas, y luego se halla la diferencia entre los ángulos ϕ de cada pareja de acelerómetros ubicada a cinco filas de distancia.

```
CartToSph = CoordinateTransformData[{"Cartesian" → "Spherical", 3} , "Mapping"];

sph = CartToSph /@ (aux);
angulos = (Take[Transpose[sph][[3]], {7, 22}] - Take[Transpose[sph][[3]], {2, 17}]);
```

Figura 3.4 Obtención de los ángulos usando coordenadas esféricas.

3.1.4. Cálculo de ángulos mediante el método de kurata

Con este método los ángulos de las articulaciones se calculan directamente de las mediciones cartesianas de los acelerómetros. En la Figura 3.5 se presenta la implementación del algoritmo para obtener los ángulos de las articulaciones, basado en el modelo de kurata.

```
Ay = Take[Transpose[aux][[2]], {2, 17}];
Az = Take[Transpose[aux][[1]], {2, 17}];
By = Take[Transpose[aux][[2]], {7, 22}];
Bz = Take[Transpose[aux][[1]], {7, 22}];
angulos1 = ArcTan[Ay By + Az Bz, -(Ay Bz - Az By)]
```

Figura 3.5 Obtención de los ángulos usando el modelo de Kurata.

3.2. Caso 2. Cálculo de ángulos en la FPGA.

A partir de las mediciones de los acelerómetros previamente centralizadas en la FPGA, se desarrolló in situ el cálculo de los ángulos, utilizando una arquitectura de procesamiento concurrente basada en el modelo propuesto por Kurata. Los ángulos calculados en la FPGA son enviados a Mathematica a tráves de una interfaz de comunicación UART.

Para el cálculo de los ángulos en la FPGA se implementó una arquitectura de procesamiento concurrente basada en el modelo propuesto por Kurata, el cual involucra el uso de una función arco-tangente. Se propuso un nuevo algoritmo para la implementación de la función arco-tangente, el cual se basa en aspectos simétricos relacionados con la distancia angular a la diagonal de las funciones tangente y cotangente. Se diseñó y se implementó en hardware un componente, basado en el algoritmo propuesto, que calcula la función arco-tangente y tiene una resolución de 1°, lo cual es suficiente para esta aplicación.

3.2.1. Diseño e implementación de la función Arco-Tangente

La función Tangente de un ángulo es la razón entre el cateto opuesto (Y) y el cateto adyacente (X) de un triángulo rectángulo. Cuando la magnitud de X es mayor a la magnitud de Y, la magnitud de la tangente es menor a 1. Por el contrario, cuando la magnitud de X es menor a la magnitud de Y, la magnitud de la tangente es mayor a 1. Cuando la magnitud de X es igual a la magnitud de Y, la magnitud de la tangente es 1. La función Cotangente es la razón trigonométrica inversa. Estas funciones, tangente y cotangente, se complementan de forma simétrica. Este diseño está basado en dicha simetría.

2.1.1.3.Análisis de simetría

Las funciones inversas a la tangente y a la cotangente son las funciones arco-tangente y arco-cotangente. Ambas funciones inversas permiten obtener el valor de un ángulo a partir de los valores de X y Y; el dominio de ambas funciones es todos los reales.

En la Figura 3.6 se observa que cuando la magnitud de la tangente es mayor a 1, la de la cotangente es menor a 1, y viceversa. También se observa que a partir de estas funciones se puede generar una función a trozos continua, cuyo rango está acotado a [-1, 1].

Considerando un dominio simétrico de [-180°, 180°], y teniendo en cuenta que los ángulos positivos se recorren en sentido anti-horario y los ángulos negativos en sentido horario, se encuentra que la única diferencia entre las componentes X y Y de los ángulos Ø y –Ø es el signo de Y. En la Figura 3.6 se observa que la función a trozos resultante es impar, es decir, F[Ø] = -F[-Ø]. Esto significa que se puede calcular la magnitud de los ángulos sin tener en cuenta el signo de Y; luego se utiliza el signo de Y para determinar el signo del ángulo.

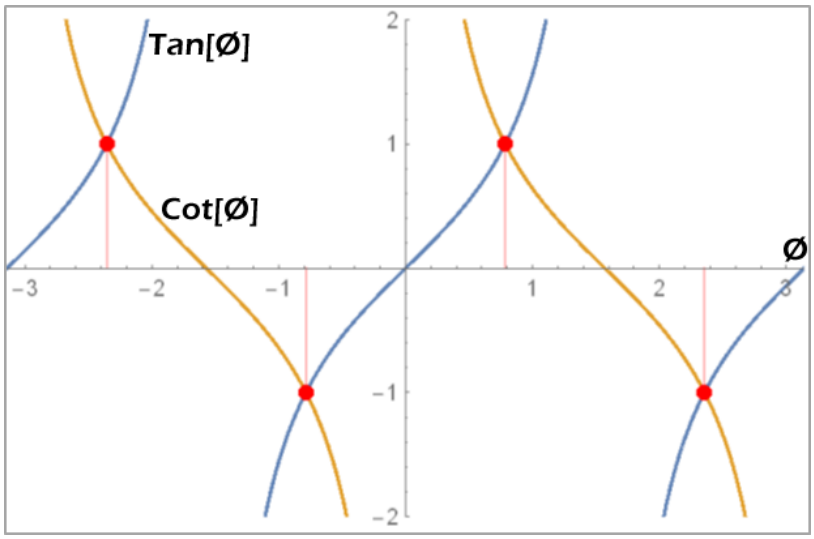


Figura 3.6 Comparación entre la función tangente (azul) y cotangente (naranja).

En la Figura 3.7 se observa una gráfica de las magnitudes de las funciones tangente y cotangente. Los puntos donde la curva toma el valor de 0 están asociados a los ángulos $n\frac{\pi}{2},\ para\ todo\ n \in Z$. Los puntos donde la curva toma el valor de 1 (rojo) están asociados a los ángulos $n\frac{\pi}{2} \pm \frac{\pi}{4},\ para\ todo\ n \in Z$. Se observa que existe una simetría par en cada una de las diagonales (puntos rojos). Es decir, $|\tan(\alpha - \beta)| = |\cot(\alpha + \beta)|$ para cualquier valor de β y para los valores de $\alpha = n\frac{\pi}{2} \pm \frac{\pi}{4},\ para\ todo\ n \in Z$.

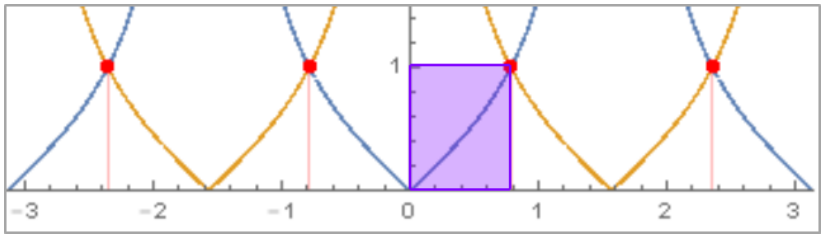


Figura 3.7 Valor absoluto de las funciones tangente (azul) y cotangente (naranja).

La función a trozos de la Figura 3.7, para una β dada, presenta el mismo valor en cada uno de los segmentos $\left[n\frac{\pi}{8}, (n+1)\frac{\pi}{8}\right]$. Eso significa que se puede determinar completamente el valor del ángulo si se conoce la distancia angular a la diagonal más cercana (DAD), la relación entre las magnitudes de X y Y (X<Y ó X>Y) y los signos de X y Y. La DAD puede tomar valores entre 0° y 45 °; El signo de Y determina el signo del ángulo; El signo de X determina si la magnitud del ángulo es mayor o menor a 90°; La relación entre las magnitudes de X y Y (X<Y ó X>Y) determina si el ángulo está más cerca del eje X (0°, 180°) o del eje Y (90°). En la Figura 3.8 se presenta el algoritmo para determinar la magnitud del ángulo.

$\text{signoX} = (+)\ y\ X > Y,$	$\text{Angulo:} = 45° - \text{DAD}$
$\text{signoX} = (+)\ y\ X \leq Y,$	$\text{Angulo:} = 45° + \text{DAD}$
$\text{signoX} = (-)\ y\ X \leq Y,$	$\text{Angulo:} = 135° - \text{DAD}$
$\text{signoX} = (-)\ y\ X > Y,$	$\text{Angulo:} = 135° + \text{DAD}$

Figura 3.8 Funciones de Post-Procesamiento para obtener la magnitud de los ángulos a partir de la DAD.

2.1.2.3.Algoritmo a implementar

En la Figura 3.9 se presenta la arquitectura de hardware utilizada para la implementación de la función arco-tangente en la FPGA, la cual está basada en una arquitectura propuesta en el 2009 [1]. Ésta arquitectura está compuesta por 5 etapas: Conversión de formato, multiplexación X Y, división, core y post-procesamiento. La función arco-tangente implementada está basada en el concepto de distancia angular a la diagonal (DAD) expuesto en el análisis de simetría y tiene una resolución de 1°.

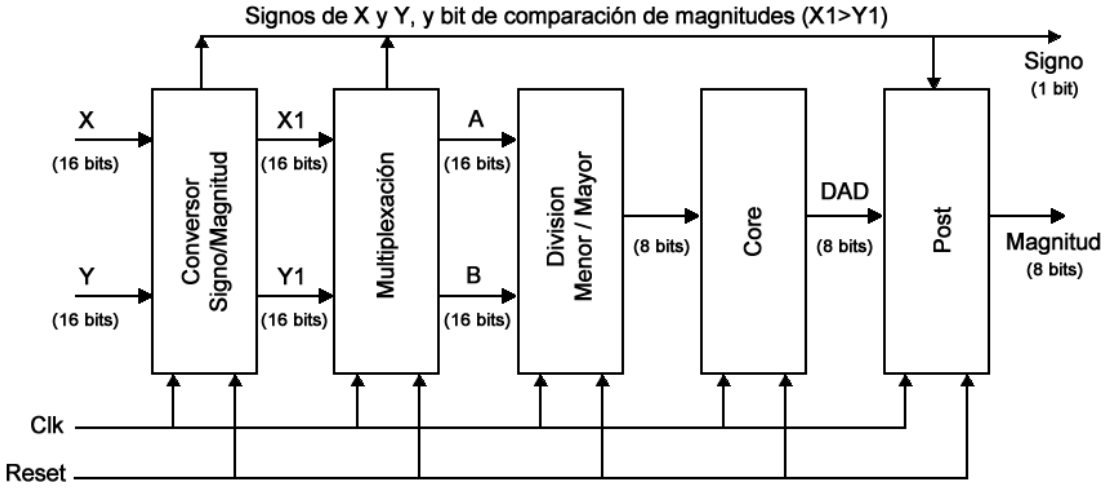


Figura 3.9 Arquitectura de la función Arco-tangente implementada.

2.1.3.3.Formato Utilizado

El algoritmo propuesto trabaja internamente en formato signo-magnitud, ya que facilita comparar la magnitud de dos números. Adicionalmente, para la transmisión de los ángulos a dispositivos externos, se diseñó una estructura de trama donde los bits de signo de los 16 ángulos calculados, son empaquetados en 2 bytes (ver Figura 3.16). La implementación en hardware de este bloque convierte los datos de complemento a 2 a signo-magnitud; los signos se obtienen a partir del bit más significativo de cada dato, mientras que la magnitud se obtiene mediante una función de valor absoluto: si X es positivo, F[x] = X. Si X es negativo, F[x] = 0 – X (ver Figura 3.11).

2.1.4.3. Multiplexación X Y

En esta etapa se multiplexan los datos para la división. Para reducir el dominio de la función a [0, 1], el numerador (A) debe ser menor que el denominador (B). El algoritmo implementado se presenta en la Figura 3.10.

$$\begin{aligned} \boldsymbol{X} > \boldsymbol{Y}, \quad & \text{entonces } A = Y, B = X \\ \boldsymbol{X} \leq \boldsymbol{Y}, \quad & \text{entonces } A = X, B = Y \end{aligned}$$

Figura 3.10 Algoritmo para identificar quien es menor entre X y Y.

2.1.5.3.División

Como el numerador (A) es menor que el denominador (B), el resultado de la división está entre 0 y 1, lo cual impide que este se pueda representar mediante números enteros. Se concatenó 8 ceros a la derecha del numerador, lo cual corresponde a una amplificación de 256, para que el resultado de la división esté entre 0 y 256. Esto permite utilizar división entera. El procedimiento de amplificación es equivalente a usar aritmética de punto fijo con n bits en la parte decimal. En la Figura 3.11 se observan las 3 etapas mencionadas: conversión de formato, multiplexación X Y y división entera.

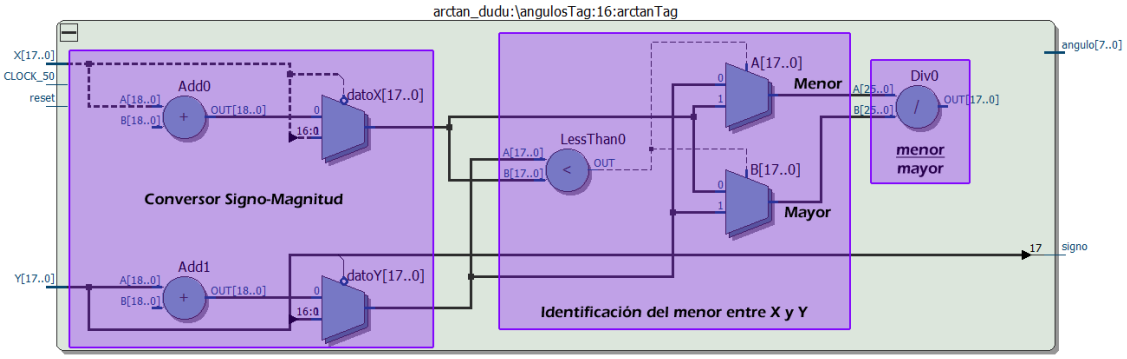


Figura 3.11 Procesos para dividir la magnitud del menor sobre la del mayor.

2.1.6.3.Distancia angular a la diagonal (DAD) - LUT de Arco-Tangente

La función Arco-Tangente implementada calcula los ángulos con una precisión de 1°. Como el rango de la DAD es de [0°, 45°], solo es necesario implementar una LUT con los valores de las tangentes de los ángulos 0.5°, 1.5°, … , 44.5°, que son los valores umbrales entre cada ángulo entero. En la Figura 3.12 se presenta esta lista de valores, resaltando en verde los valores más alejados de la diagonal, y en naranja los valores más cercanos. Estos valores están amplificados en 256, por lo que el valor de la DAD se puede determinar utilizando solo valores enteros.

$$\begin{pmatrix} 2 & 25 & 47 & 71 & 96 & 122 & 151 & 183 & 219 \\ 7 & 29 & 52 & 76 & 101 & 128 & 157 & 189 & 226 \\ 11 & 34 & 57 & 81 & 106 & 133 & 163 & 196 & 235 \\ 16 & 38 & 61 & 86 & 111 & 139 & 169 & 204 & 243 \\ 20 & 43 & 66 & 91 & 117 & 145 & 176 & 211 & 252 \end{pmatrix}$$

Figura 3.12 Tabla con los valores umbrales utilizados para determinar la DAD.

Para calcular la DAD se compara el cociente entero de la división (menor/mayor) con cada uno de los elementos de la tabla. Indicando con un '1' si dicho cociente es mayor al valor de la tabla. Si la distancia angular es 0°, el resultado es una fila con 45 ceros; Si es 15°, una fila con 30 ceros y 15 unos; y si es 45°, una fila con 45 unos. El resultado es una tabla binaria de 45 columnas por 46 filas, donde la primera fila está asociada a la distancia angular 0° y la última fila a la distancia angular 45° $\left(\frac{\pi}{4}\right)$. Esta tabla se debe decodificar para obtener una representación de los datos en binario decimal. La decodificación directa de la matriz diagonal a decimal requiere el uso de muchos recursos lógicos, siendo más eficiente hacer una decodificación previa a una matriz one-hot.

Esta decodificación previa se hace mediante un arreglo de compuertas XOR, las cuales comparan cada pareja de bits consecutivos, ya que el valor del ángulo está asociado a la posición donde se presenta la transición de 0 a 1. Si dicha transición no se presenta, que es el caso de una fila de ceros (DAD=0) o una fila de unos (DAD=$\frac{\pi}{4}$), el ángulo se detecta mediante una NOR entre los 2 LSb o mediante una AND entre los 2 MSb. El resultado es una matriz cuadrada de 46 bits que codifica en One-Hot los ángulos de 0 a $\frac{\pi}{4}$, como se observa en la Figura 3.13. Finalmente se implementa un codificador de One-Hot a Decimal para obtener la magnitud de la DAD en representación decimal.

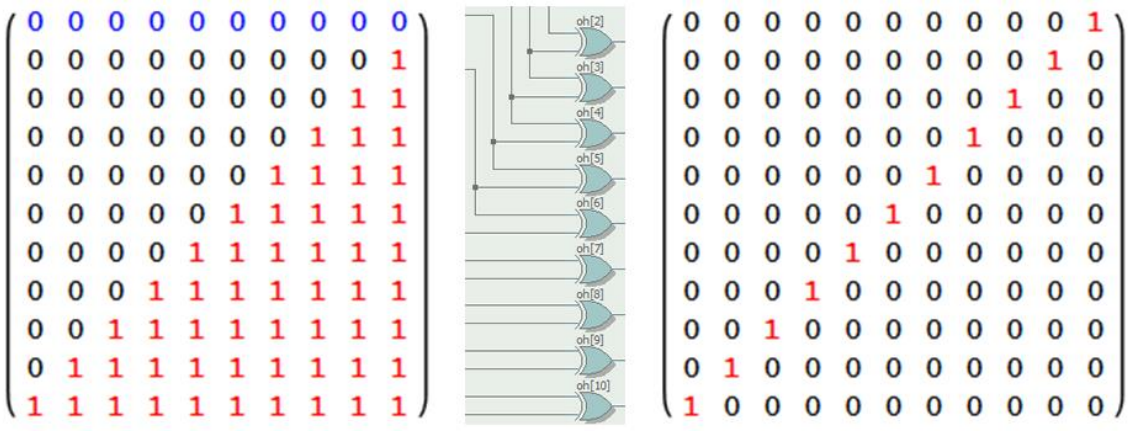


Figura 3.13 Conversión de matriz triangular a matriz diagonal mediante un vector XOR y una AND.

2.1.7.3.Cálculo de ángulos a partir de la distancia angular a la diagonal

En la Figura 3.8 se presentó el algoritmo para determinar la magnitud del ángulo a partir de la distancia a la diagonal, la relación entre las magnitudes de X y Y (X>Y o Y>X) y el signos de X. El signo de Y determina el signo del ángulo. En la Figura 3.14 se presenta la implementación en hardware de este algoritmo. Se efectúan de forma concurrente las 4 operaciones aritméticas del algoritmo, y mediante el uso de multiplexores se selecciona el resultado de la operación aritmética adecuada. Los bits de selección de los multiplexores son el bit de signo de X y el resultado booleano de la comparación X > Y.

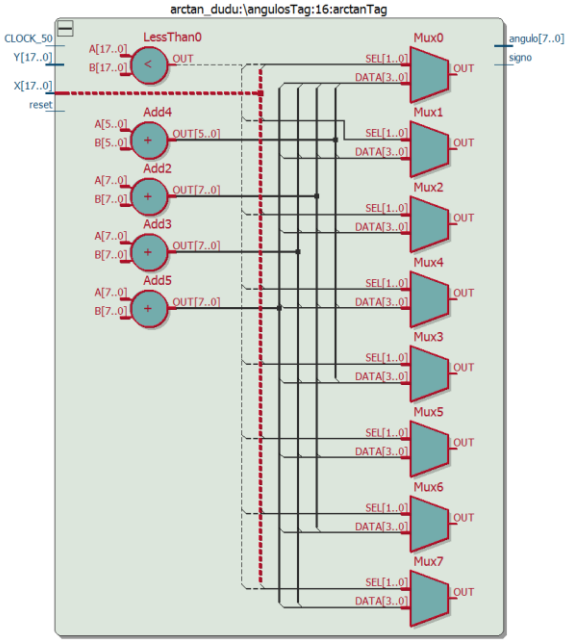


Figura 3.14 Obtención de los ángulos a partir del signo de X, de la comparación X>Y y de la DAD.

3.2.2. Cálculo de los ángulos presentes en las articulaciones

La función Arco-Tangente se implementó con el propósito de calcular los ángulos presentes en las articulaciones, mediante el modelo propuesto por Kurata. Para la implementación en hardware del modelo propuesto por kurata, los datos se organizan para facilitar la implementación de las operaciones posteriores, al igual que el algoritmo implementado en Mathematica: se replica 4 veces la señal asociada al acelerómetro de la palma y se reorganizan internamente los datos de los ejes del acelerómetro presente en la DE0-nano.

Luego se procesan los datos mediante el modelo matemático propuesto por kurata. En la Figura 3.15 se presenta el RTL de la implementación en hardware del modelo de Kurata. El operando Y de la tangente corresponde a una resta de productos, mientras que el operando X a una suma de productos. Se implementó un registro a la salida el cual se actualiza sincrónicamente con el tiempo de muestreo.

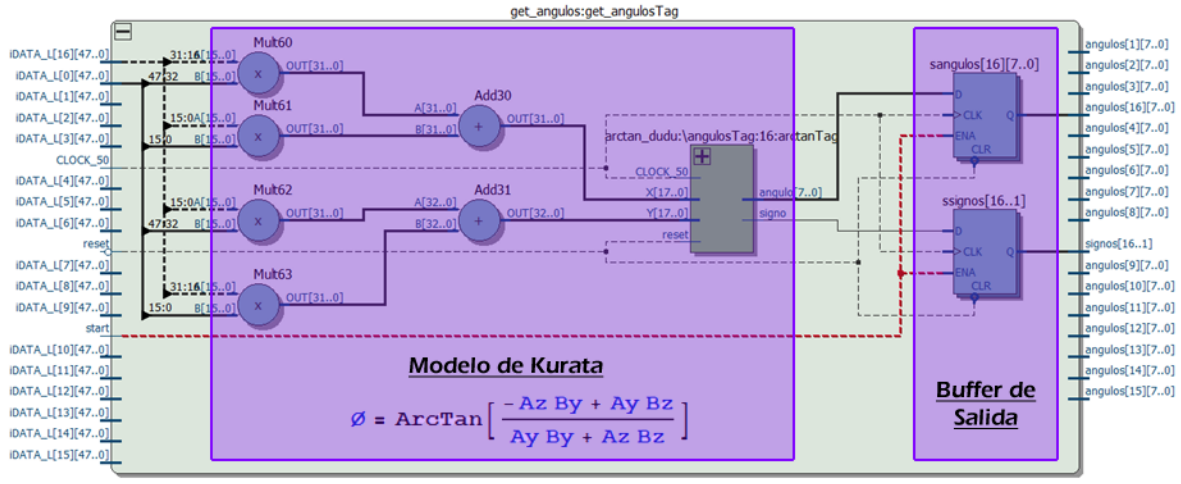


Figura 3.15 Implementación en hardware del modelo de kurata.

3.2.3. Trama transmitida

Como los ángulos son calculados en la FPGA, la trama transmitida es solo de 24 bytes: 6 bytes de cabecera, 16 con las magnitudes de los ángulos y 2 con los signos (16 bits), como se indica en la Figura 3.16. Los datos son transmitidos con el fin de permitir a dispositivos externos, como un PC, hacer uso de la información generada por el guante electrónico. También se observa que la trama es mucho más pequeña que la utilizada para calcular los ángulos de forma remota, la cual es de 108 bytes.

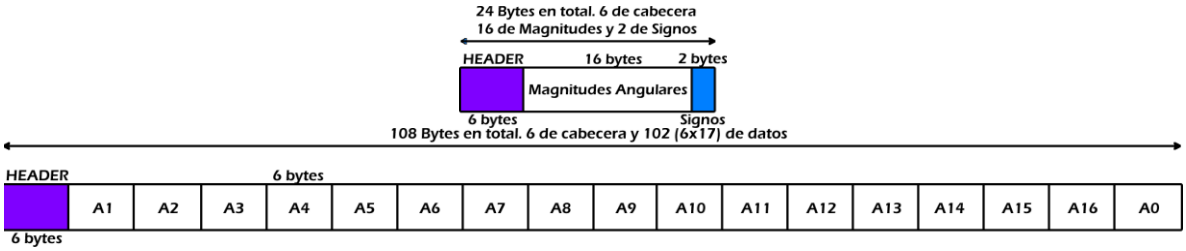


Figura 3.16 Trama utilizada para transferir los ángulos en formato Signo – Magnitud.

3.2.4. Adquisición y organización de los datos en Mathematica

El proceso de adquisición es idéntico al mencionado en el Caso 1 (3.1.2.). Luego de adquirir la trama, los datos deben ser ensamblados aplicando a cada magnitud su correspondiente signo, como se indica en la Figura 3.17. Para ensamblar la magnitud y el signo se utilizó una instrucción aritmética basada en listas, lo cual es más eficiente que el uso de condicionales. El cero representa un signo positivo y el uno un signo negativo; al multiplicar por -2 el bit de signo y sumarle 1 al resultado, el signo positivo queda asociado al valor 1 y el signo negativo al valor -1; esto permite ensamblar la lista de magnitudes con

la lista de signos mediante una multiplicación. Finalmente, los ángulos son pasados a radianes, formato en que las funciones de Mathematica interpretan los ángulos.

```
RunScheduledTask[
  bufferAux = AppendTo[bufferAux, DeviceReadBuffer[xbee]] // Flatten;
  i = seqpos[bufferAux, {170, 85, 85, 85, 85, 170}];
  If[Length[i] == 0, i = {1}];
  bufferAux = Drop[bufferAux, i[[1]] - 1];
  If[Length[bufferAux] ≥ 24,
   trama = Take[bufferAux, 24]; bufferAux = Drop[bufferAux, 24];
  ];
  mangulos = trama[[7 ;; 22]];
  signos = trama[[23 ;; 24]];
  angulos = -Reverse[Flatten[2 IntegerDigits[signos, 2, 8] - 1]] * mangulos * π/180;
  , 0.01
 ];|
```

Figura 3.17 Algoritmo utilizado para la adquisición de las tramas con los ángulos.

3.3. Análisis de casos: Procesamiento Externo vs Local

El procedimiento para obtener los ángulos es prácticamente el mismo para ambos casos: A partir de las mediciones de inclinación se obtienen ángulos mediante la aplicación de un algoritmo. Sin embargo, en el caso 1 (Externo) parte del procesamiento se realiza en el PC mientras que en el caso 2 se desarrolla todo en la FPGA (Local). La obtención de los ángulos de forma local representa autonomía, ya que el guante electrónico no depende de sistemas externos para el cálculo de los ángulos. Cuando se transfieren las mediciones de los acelerómetros, es necesario que en el dispositivo externo se implementen los algoritmos que permitan calcular los ángulos presentes en las articulaciones.

El procesamiento externo es muy importante en la fase de diseño, ya que se transfieren al PC todas las mediciones de los acelerómetros. Esto permite analizar los datos, probar algoritmos, etc., sin presentar tiempos de desarrollo muy altos. Sin embargo, el cálculo en tiempo real de los ángulos genera una alta carga de procesamiento. Utilizando un procesador Intel core 2 duo con 4 GB de RAM, el máximo tiempo de procesamiento continuo fue cercano a 25s pero lo típico fue de 10s. Con un procesador Intel i7 con 16GB de RAM se alcanzó un tiempo de procesamiento continuo cercano a 90s y típico de 30s. Al calcular los ángulos de forma local, se liberó al dispositivo externo de esta carga de procesamiento, lo que mejoró el rendimiento general del sistema y permitió monitorear la postura de la mano de forma continua a través de su representación visual.

El guante electrónico se comunica con dispositivos externos a tráves de una interfaz de comunicación UART, la cual se configuró a 115200 Kbps. Se utilizaron módulos RF XBeePro, configurados en modo transparente, para llevar a cabo la comunicación UART de

forma inalámbrica (Wireless), los cuales tienen un throughput máximo de 35Kbps. En consecuencia, cuando los ángulos se calculan localmente se alcanzan ratas de muestreo de 100Hz, ya que se utilizan tramas de 24 bytes. Cuando se calculan en un dispositivo externo, se utilizan tramas de 108 bytes, alcanzando ratas de muestreo de 25Hz. Al calcular los ángulos de forma local se utilizan tramas 4.5 veces más pequeñas y se alcanzan ratas de muestreo 4 veces más altas que las presentadas para el cálculo de ángulos en un dispositivo externo.

3.4. Recursos de hardware utilizados

En la tabla 3.1 se presenta un resumen de los recursos lógicos utilizados por los componentes del sistema implementado en la FPGA. En la tabla se observa que el sistema de cálculo de ángulos utiliza el 77% de los elementos lógicos (LEs) y el 100% de los multiplicadores embebidos. Se observa que el uso de multiplicadores embebidos no se debe a la función arco-tangente implementada, la cual hace parte del sistema de cálculo de ángulos. El resto de lógica utilizada en la FPGA (glue lógic) corresponde a la interfaz multi I/O SPI, la interfaz UART y las diferentes unidades de control presentes en el sistema.

Componente	Elementos lógicos	Registros (Flip-Flops)	Mult. de 9bits
Arc-Tan	1039 (5%)	0	0
Kurata x 16	17216 (77%)	144 (1%)	128 (100%)
Glue logic	1863 (8%)	2479 (11%)	0
Total	20118 / 22300 (90%)	2623 / 22300 (12%)	128 / 128 (100%)

Tabla 3.2. Resumen de recursos lógicos utilizados por los principales componentes del sistema

El sistema de cálculo de ángulos se puede optimizar mediante la implementación de una arquitectura pipeline. Dicha arquitectura incrementa el uso de registros pero a cambio, reduce el uso de multiplicadores embebidos a un 6% y reduce el uso de recursos lógicos más o menos al 10%.

3.5. Representación visual de la mano

El sistema de verificación del funcionamiento del guante electrónico es una representación visual de la mano. Este sistema permite visualizar en tiempo real la postura de la mano a través de un modelo virtual 3D. En los objetivos del proyecto se acota el problema del reconocimiento de la postura de la mano a rotaciones sobre un plano vertical y considerando un grado de libertad por articulación. La implementación de la representación visual de la mano se desarrolló en Wolfram Mathematica, basada en un modelo virtual de la mano que se encuentra en las demostraciones de Wolfram [2].

La implementación de este proceso de verificación se divide en 4 etapas. La primera etapa consiste en establecer el modelo de la mano a representar. La segunda etapa consiste en la creación de objetos 3D con los cuales representar cada uno de los segmentos de la mano considerados en el modelo. La tercera etapa consiste en el ensamble de estos segmentos, formando la estructura cinemática de la mano. La cuarta etapa consiste en asociar los ángulos de las articulaciones del modelo 3D con las mediciones angulares realizadas por el guante electrónico.

3.5.1. Modelo de la mano

El modelo de la mano a utilizar está compuesto por 17 segmentos y 16 articulaciones. Un segmento es un conjunto de elementos que está enlazado con otros segmentos a través de articulaciones con alto grado de movilidad. Los segmentos considerados son: Las catorce falanges, el metacarpiano del pulgar, el conjunto de los 12 huesos que componen la palma de la mano y el antebrazo. Para obtener la curvatura de la mano, la palma se implementa mediante los segmentos asociados a sus 4 metacarpianos.

Las 16 articulaciones consideradas son: Las 9 articulaciones interfalangicas, las 5 metacarpofalángicas, la carpo-metacarpiana del dedo pulgar y la muñeca. Las articulaciones metacarpofalángicas son de 2 grados de libertad (2 DOF), a excepción de la presente en el dedo pulgar (1 DOF), y consideran solo los movimientos de flexión y extensión. La articulación carpo-metacarpiana también es de 2 DOF y se consideran solo los movimientos de aducción y abducción radial. La muñeca es 2 DOF y se consideran solo los movimientos de flexión y extensión.

3.5.2. Creación de los segmentos de la mano con objetos virtuales 3D

Los objetos 3D que representan los segmentos de la mano se crean al inicio de la aplicación, y luego se procede a su ensamble. En la Figura 3.18 se presentan las instrucciones utilizadas para la creación de los segmentos de la mano. Primero, se definen los valores con los cuales se ajusta la escala de la longitud de cada dedo. Luego, se calculan las longitudes de cada segmento, asumiendo que las longitudes de los segmentos de cada dedo están relacionadas con la sucesión de Fibonacci.

```
(*Definir dimensiones*)
dimPulgar = 55; dimIndice = 45; dimCorazon = 48; dimAnular = 45; dimMeñique = 40; rg = 2; rf = 20;
longBase = {dimMeñique, dimAnular, dimCorazon, dimIndice, dimPulgar};
dimPiezas = ({2/8, 3/8, 5/8, 1} x /. x → longBase // Flatten) y /. y → {0, 0, 1}; dimPiezas[[20]] /= 2;
(*Crear piezas de la mano*)
piezas = Table[Cuboid[-{rg, rg, 0}, dimPiezas[[i]] + {rg, rg, 0}], {i, 20}];
ensamble = Range[20];
ensamble[[1 ;; 5]] = piezas [[1 ;; 5]];
```

Figura 3.18 Creación de los segmentos de la mano.

En la variable piezas se almacenan los segmentos 3D, modelados mediante cubos con las longitudes previamente mencionadas y con un área que asemeje la proporción ancho-longitud de los segmentos de la mano. En la Figura 3.19 se presenta la forma en que están distribuidos estos 20 segmentos. Finalmente, en la variable "ensamble" se almacenan los diferentes pasos del ensamble de los objetos; se deben llenar las primeras posiciones con los segmentos asociados a las falanges distales, lo que facilita el proceso posterior de ensamblado.

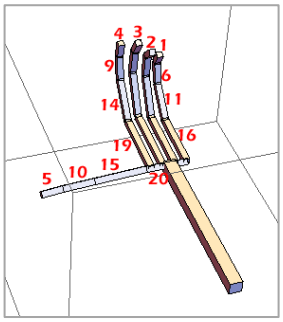


Figura 3.19 Numeración de los segmentos utilizados para modelar la mano.

3.5.3. Ensamble de los segmentos 3D y asociación con los ángulos medidos

En la variable piezas se almacenan los segmentos 3D que deben ser ensamblados. El proceso de ensamble se basa en operaciones de rotación y traslación: como se observa en la Figura 3.20, el cual se puede entender a través del caso del dedo índice.

```
RunScheduledTask[
  angulo = angulos1;
  Do[ensamble[[5 + i]] = {Translate[Rotate[ensamble[[i]], angulo[[i]], {0, 1, 0}], dimPiezas[[5 + i]]], piezas[[5 + i]]}, {i, 1, 15}];
  mano = {
    Translate[ensamble[[16]], {0, -3/5 rf, 0}], Translate[ensamble[[17]], {0, -1/5 rf, 0}],
    Translate[ensamble[[18]], {0, 1/5 rf, 0}], Translate[ensamble[[19]], {0, 3/5 rf, 0}],
    Translate[Rotate[Rotate[ensamble[[20]], π/4, {0, 0, 1}], -π/2, {1, 0, 0}], {0, -3/5 rf, 0}]
   };
  brazo = {Translate[Rotate[mano, angulo[[16]], {0, 1, 0}], 4 dimPiezas[[20]]], Cuboid[-2 {rg, rg, 0}, 4 dimPiezas[[20]] + 2 {rg, rg, 0}]};
  , 0.04
];
```

Figura 3.20 Ensamble de los segmentos 3D y asociación con las mediciones angulares.

El dedo índice está asociado a las piezas 4, 9, 14 y 19, correspondientes a la falange distal (A), medial (B), proximal (C) y al metacarpiano (D), respectivamente. Primero se ensamblan las falanges distal y medial (ensamble AB). Luego se une el ensamble AB con la falange proximal, formando el dedo (ensamble ABC). Finalmente, se une el ensamble ABC con el metacarpiano, formando el ensamble ABCD. El proceso es el mismo para cada uno de los dedos.

Para unir los segmentos A y B, se rota el segmento A y luego se traslada al extremo del segmento B. Para unir el segmento AB con el C, primero se rota el segmento AB y luego se traslada al extremo del segmento C. Para unir el segmento ABC con el D, primero se rota el segmento ABC y luego se traslada al extremo del segmento D. Los valores que indican cuánto se debe rotar en cada articulación se almacenan en la variable angulo.

Para que la representación visual de la mano sea dinámica, el proceso de ensamblado también debe serlo. Esto se consigue mediante la configuración de un proceso periódico de ensamblado. En la Figura 3.20 se observa que, para la representación visual de la mano, tanto la actualización de los ángulos asociados a la postura de la mano como el proceso de ensamble se repiten cada 40ms (25Hz).

3.5.4. Visualización dinámica de la postura de la mano

La función Graphics3D se utiliza para la visualización de los modelos 3D previamente descritos. Sin embargo, esta visualización es estática. La función Dynamic es utilizada para la representación dinámica de datos que pueden cambiar con el paso del tiempo; en este caso, los datos que está representando corresponden al ensamble 3D del modelo de la mano, el cual se modifica cada vez que las mediciones transferidas por el guante cambien.

En la Figura 3.21 se presentan dos modelos 3D para la representación visual de la mano; sin embargo, en el modelo de la izquierda, la palma de la mano se representa mediante el conjunto de metacarpianos, lo cual permite modelar la curvatura de la mano. Se utilizó el modelo de la izquierda como representación visual de la mano, ya que representa mejor la postura de la mano. En la Figura 3.22 se presenta una serie de imágenes donde se observa como dicha representación visual puede seguir el movimiento de la mano en tiempo real.

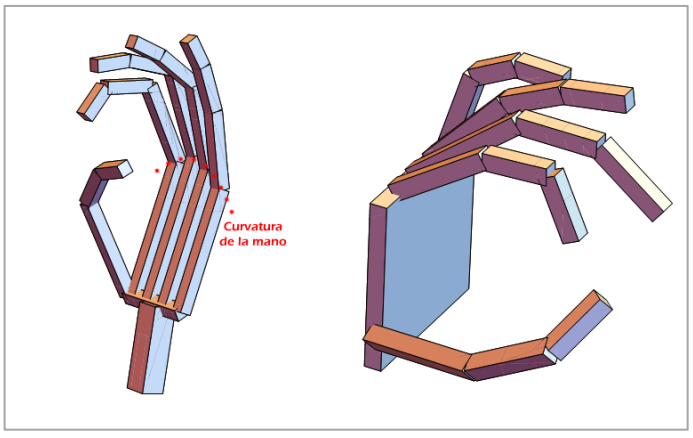


Figura 3.21 Dos modelos virtuales que sirven como representación visual de la mano.

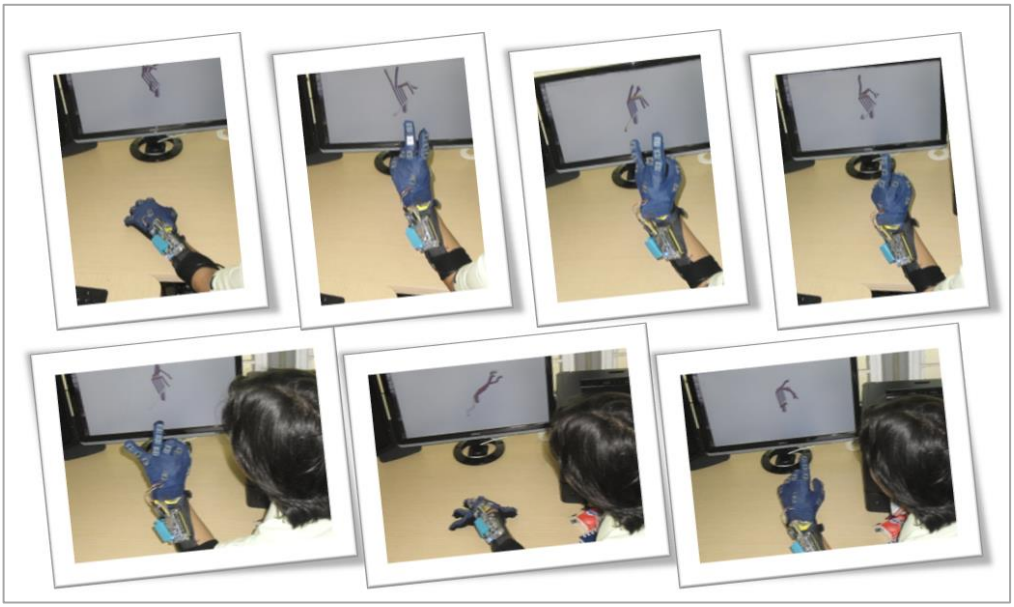


Figura 3.22. Representación visual de la mano siguiendo los movimientos de la mano.

3.6. Referencias

[1] M. saber, Y. Jitsumatsu, and T. Kohda, "A low-power implementation of arctangent function for communication applications using FPGA," in *2009 Fourth International Workshop on Signal Design and its Applications in Communications*, 2009, pp. 60–63.

[2] S. Cabal, "Hand Model," *Wolfram Demostrations*, 2007. [Online]. Available: http://demonstrations.wolfram.com/HandModel/. [Accessed: 11-Jun-2015].

Capítulo 4
Construcción del DataGlove

Un proceso esencial de este proyecto es el diseño e implementación de un sistema electrónico vestible que permita obtener los ángulos presentes en las articulaciones de la mano. Inicialmente se describen las especificaciones técnicas de los componentes electrónicos utilizados en el guante electrónico. Luego se describe la distribución de dichos componentes y la forma en que estos son interconectados. Posteriormente se indican los detalles de la implementación del sistema electrónico vestible. Finalmente, se presenta el proceso de verificación del DataGlove.

4.1. Aspectos Técnicos de los Componentes Utilizados

4.1.1. Acelerómetro ADXL345

Este sensor es usado en este proyecto, a través del módulo presentado en la Figura 4.1 (20mm x 14mm x 3mm @ 6 gramos), para medir la inclinación de la superficie dorsal de la palma de la mano, del antebrazo y de cada uno de los 3 segmentos que componen cada dedo. El ADXL345 [1] es un pequeño (3mm x 5mm x 1mm) acelerómetro digital tri-axial MEMS desarrollado por Analog Devices. Su rango de medición es de ±16 gravedades y su máxima resolución es de 4mg/LSb. Los datos están en formato complementos a 2 y se pueden consultar mediante SPI (3 o 4 hilos) o una interfaz digital I2C.

Figura 4.1 Módulo con ADXL345.

El ADXL345 es un sistema de bajo consumo de potencia. Polarizado a 3.3v, tiene un consumo de 100nA en stand-by, lo que permite optimizar el consumo de potencia cuando el dispositivo está en reposo, mientras que en modo de medición el consumo es de 180µA para tasas de muestreo superiores a los 100Hz. En los pines de entrada de la interfaz SPI la corriente es de 100nA, en los pines de salida puede llegar a 4mA (fan-out). También hay

otros 2 aspectos técnicos que se deben tener en cuenta al trabajar con este dispositivo, y son la tensión de alimentación y la tasa de muestreo.

En la Figura 4.2 se observa que el consumo de corriente incrementa de forma lineal con la tensión de alimentación, lo que implica que el consumo de potencia incrementa de forma cuadrática y es mínima cuando la tensión es la mínima soportada (2V). Sin embargo, el nivel de ruido disminuye al incrementar la tension, por lo tanto, se debe elegir si se desea un bajo consumo de potencia o un bajo nivel de ruido en la medición.

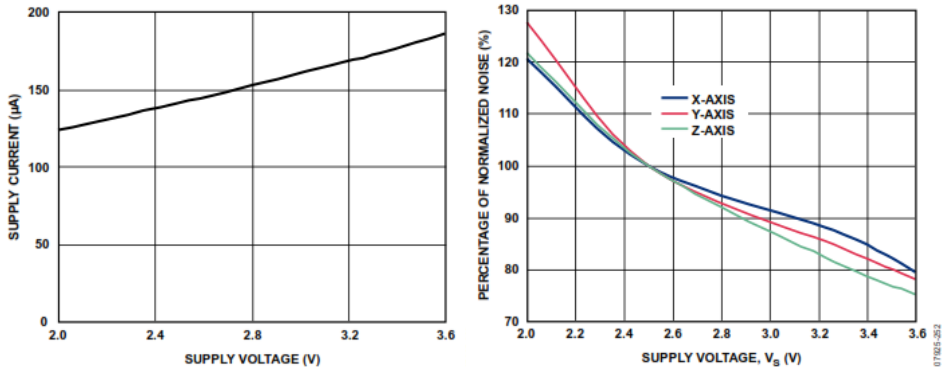


Figura 4.2 Efectos de la tensión de alimentación en el consumo de corriente y el nivel de ruido

En la Figura 4.3 se observa que tanto el nivel de ruido como el consumo de corriente es menor para tasas de muestreo bajas. La corriente nominal se presenta para tasas de muestreo superiores a 100Hz pero disminuye de forma cuadrática o exponencial para tasas de muestreo más bajas. Precisamente a partir de los 100 Hz se empieza a incrementar el nivel de ruido en las mediciones, llegando a afectar aproximadamente cinco LSb a los 3200Hz, lo que equivale a niveles de ruido de ±13%.

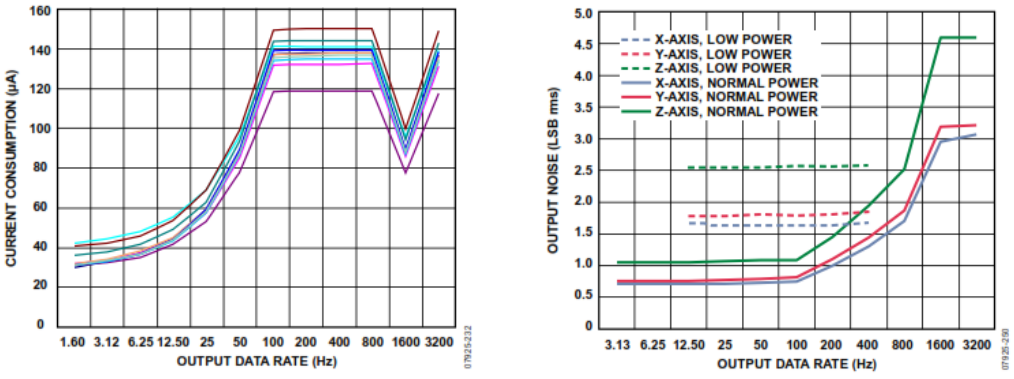


Figura 4.3 Efectos de la frecuencia de muestreo en el consumo de corriente y el nivel de ruido

4.1.2. Módulos XBEE

Los XBee son módulos de comunicación inalámbrica desarrollados por Digi International. Utilizan el protocolo de red IEEE 802.15.4 para crear redes punto a multipunto o redes punto a punto, diseñados para aplicaciones que requieren baja latencia y una sincronización de comunicación predecible [2]. A la izquierda de la Figura 4.4 se observa el módulo de comunicación XBee utilizado en este proyecto y a la derecha se observa la base usada para configurar el módulo Xbee (USB) y como interfaz entre el módulo XBee y la FPGA (UART).

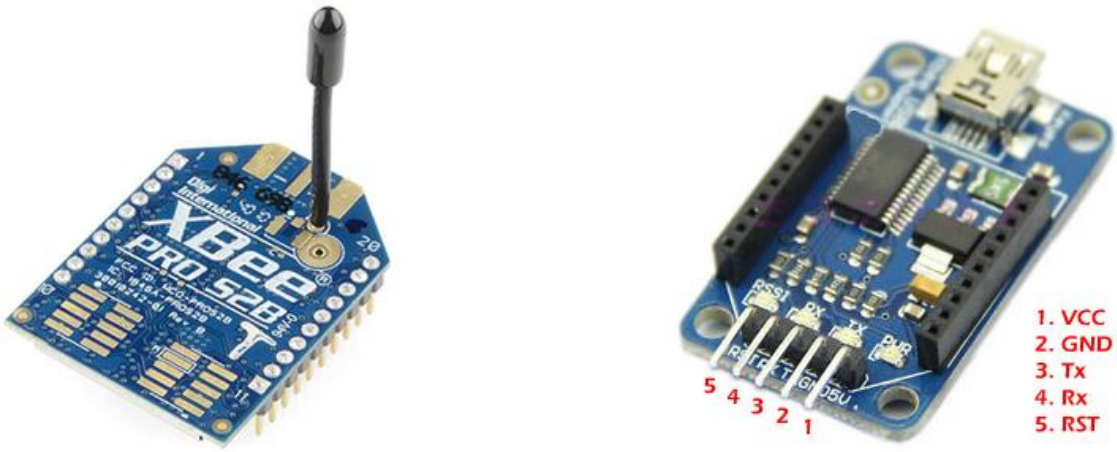


Figura 4.4 Módulo de comunicación XBee PRO S2B y su base para conexión y programación.

El XBee PRO S2B se polariza en el rango de 2V a 3.6V. Para 3.3V presenta un consumo de 15mA en stand-by y de 132mA en modo de transmisión [3]. La tasa de bits RF del XBee PRO S2B es 250Kbps, lo que permite un máximo throughput de 35Kbps. La máxima tasa de bits de la interfaz serial es 1Mb/s; normalmente se configura con ratas de baudios estándar del puerto serial, por ejemplo 9600bps. Finalmente, la potencia de transmisión de este módulo es ajustable hasta 63mW, garantizando un alcance de 3200m en línea de vista y de 90m en interiores.

4.1.3. Tarjeta de desarrollo DE0-nano

La DE0-nano [4] es una plataforma de desarrollo basada en FPGA elaborada por Terasic. Es liviana y compacta (49mm x 75mm y 100g), siendo apta para la implementación de proyectos “portables”. Entre sus principales características está el uso de una FPGA Cyclone IV EP4CE22F17C6N como sistema principal de procesamiento (22320 elementos lógicos, 132 multiplicadores embebidos de 9 bits, 600Kbits de RAM embebida y 4 PLLs), 81 pines de propósito general, 7 entradas de reloj, 8 entradas analógicas y un acelerómetro ADXL345, el cual está comunicado mediante una interfaz SPI de 3 hilos. En la Figura 4.5 se presenta la forma en que están distribuidos los recursos.

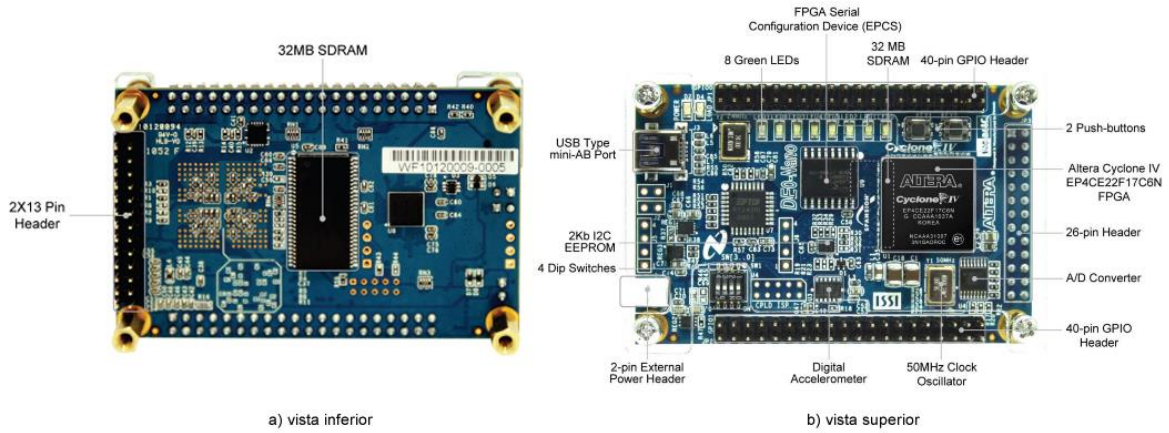


Figura 4.5 Recursos de la DE0-nano.

Cada pin GPIO de la DE0-nano tiene una capacidad de 300µA, mientras que los pines de potencia de los GPIO Expansion Header tienen una capacidad de 200mA [4]. Debido a que cada acelerómetro ADXL345 consume 180 µA polarizado a 3.3v, un pin GPIO puede alimentar un solo acelerómetro mientras que los pines de potencia pueden suministrar energía a más de 1000 acelerómetros ADXL345. El módulo XBee se debe conectar a un pin de potencia dedicado, debido al consumo de 130mA en el estado de transmisión.

En la Figura 4.6 se presenta la distribución de pines de los GPIO de la DE0-nano y su asociación con los pines de la FPGA. La DE0-nano no tiene un circuito de protección para los pines GPIO; el uso de tensiones superiores a los 4.2V reduce el tiempo de vida de la FPGA [5], por lo tanto debe evitarse.

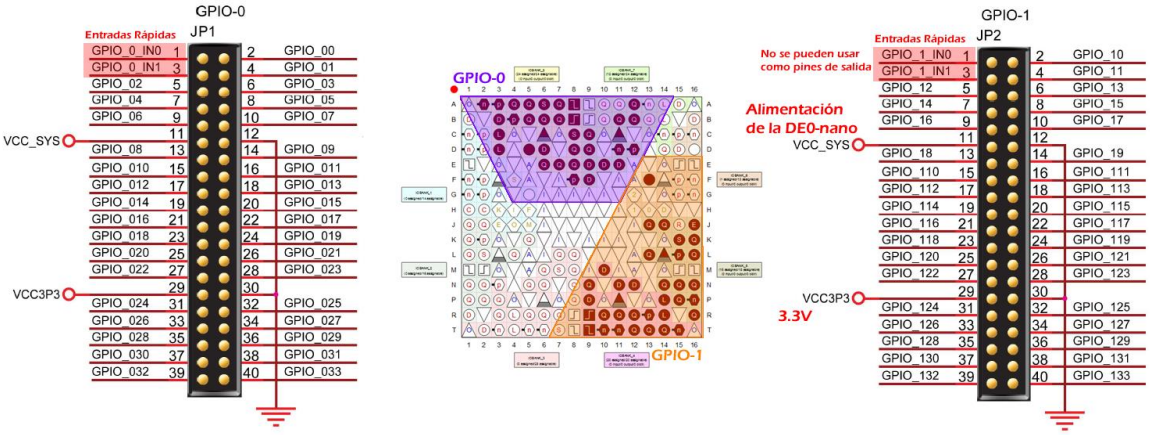


Figura 4.6 Distribución de los pines correspondientes a los 2 GPIO Headers de 40 pines.

Como el grupo de investigación Bionano de la EIEE de la Universidad del Valle tiene doce DE0-nano, no fue necesario realizar una nueva compra para este proyecto.

4.2. Fabricación del Guante Electrónico

El proceso de fabricación del guante electrónico se divide en tres etapas. La primera es la fabricación del guante textil, la segunda corresponde a la distribución e instalación del sistema electrónico, y la tercera etapa al acople del sistema electrónico con el guante textil. Los procesos de verificación y ajuste se mencionan en la sección 4.3.

4.2.1. Fabricación del guante textil

Para determinar la postura de la mano es necesaria la instalación de un sistema de medición electrónico sobre un guante textil, con el fin de obtener adecuadamente la inclinación de la superficie dorsal de los segmentos de la mano. Para esto, es necesario el uso guantes que se adapten al contorno de la mano, y al mismo tiempo, sean aptos para acoplarse al sistema electrónico requerido, y es difícil de encontrar en el mercado guantes con estas características.

"theglovesproject" es un sitio web dedicado al desarrollo de guantes de datos, donde se pueden encontrar moldes de guantes que se adaptan muy bien al contorno de la mano[6]. En la Figura 4.7 se presentan los moldes mencionados, los cuales fueron adaptados a las necesidades del diseño (talla, puntas cerradas).

En la Figura 4.8 se presentan los moldes utilizados y el guante textil fabricado. Todo el proceso de fabricación del guante textil fue desarrollado por alguien con experiencia en diseño y confección. En la parte superior se utilizó licra doble punto azul, la cual es elástica en ambas direcciones y tiene orificios que mejoran el proceso de ventilación de la mano, reduciendo la transpiración. En la parte inferior se utilizó franela licrada negra, la cual es elástica a lo ancho, mejorando la adaptabilidad del guante al contorno de la mano.

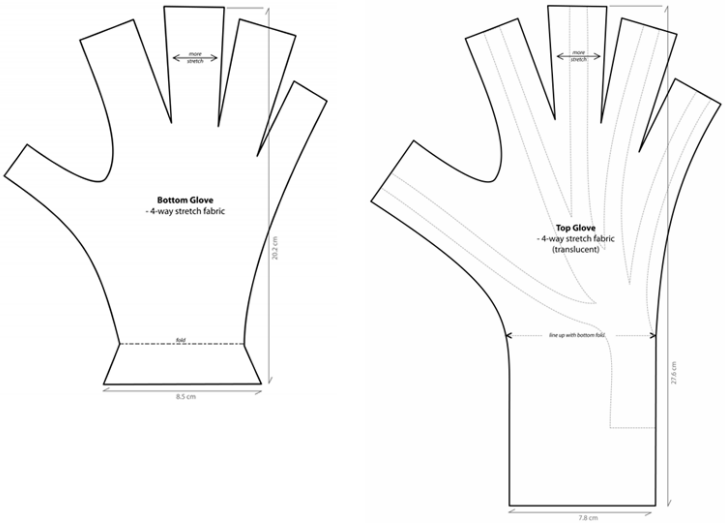


Figura 4.7 Molde de guante utilizado en este proyecto.

Figura 4.8 Proceso de fabricación del guante textil.

4.2.2. Instalación del sistema electrónico

Después de fabricar el guante textil, se procede a la instalación del sistema electrónico, cuyo diseño se basó en dos aspectos: mostrar los acelerómetros y ocultar el cableado. Para lo cual, se cortó dos veces el molde asociado a la parte dorsal de la mano (Azul), se instalaron los sensores sobre la superficie del molde exterior y se ocultaron los cables entre las dos capas de tela. En la Figura 4.9 se presenta la distribución de los acelerómetros sobre el molde de tela y la asociación que deben tener con los segmentos de la mano con el fin de obtener la postura de la mano.

Luego de instalar los acelerómetros sobre el molde de tela se procede al cableado eléctrico. Este cableado se usa para centralizar en la FPGA las mediciones de los acelerómetros, a través de una interfaz multi I/O SPI. Se utilizó cable calibre 24 AWG de múltiples hilos, ya que ofrece mayor durabilidad y flexibilidad que los alambres.

Para conectar el Xbee y los acelerómetros del guante a la DE0-nano, se utilizó un solo GPIO Expansion Header, dejando libre el otro para futuras expansiones. En la Figura 4.10 se presenta la distribución de las conexiones de los acelerómetros y el XBee sobre los pines del GPIO Expansión Header. Con el fin de reducir el fan-out de los pines de la FPGA,

correspondiente a las señales CS y CLK, se estableció un CS y un CLK para cada dedo. Los 16 acelerómetros se polarizan con los pines 11 y 12. Cada puerto de datos (SDIO) de la interfaz multi I/O SPI se cablea de forma independiente. En la Figura 4.11 se presenta el sistema electrónico cableado.

Figura 4.9 Distribución de los acelerómetros en la mano.

39	37	35	33	31	29	27	25	23	21	19	17	15	13	11	9	7	5	3	1
40	38	36	34	32	30	28	26	24	22	20	18	16	14	12	10	8	6	4	2

Los pines 1 y 3 son entradas rápidas, así que no pueden usarse como salidas
Los pines 11 y 12 son los pines usados para la alimentar los acelerómetros (5V)
La señal CS se transmite a través de los pines 2, 4 6, 8 y 10.
La señal CLK se transmite a traves de los pines 5, 7, 9, 33 y 34.
Las señales SDIO se conectan a través 16 pines, del pin 13 al pin 28
Los pines 29 y 30 son los pines usados para la alimentar el XBee (3.3V)
Las señales Tx y Rx del XBee se conectan a través de los pines 31 y 32

Figura 4.10 Distribución de las conexiones de los acelerómetros y el XBee sobre los pines del GPIO.

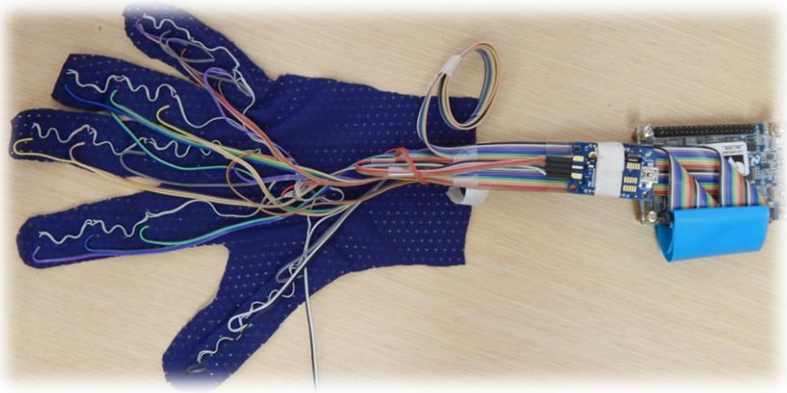


Figura 4.11 Cableado del sistema electrónico.

4.2.3. Acople del sistema electrónico con el guante textil

Una vez verificado el correcto desempeño del cableado y del sistema electrónico, se procede al acople con el guante textil. Este se logra cosiendo el molde sobre el que está distribuido el sistema electrónico al guante textil, lo cual se hace a mano ya que la presencia de los componentes electrónicos impide el uso de la máquina de coser. En la Figura 4.12 se presenta el guante electrónico acoplado y el brazalete donde se instaló la DE0-nano. El brazalete mejora el diseño porque:

- ✓ No es necesario portar en la mano la DE0-nano, mejorando la comodidad del guante.
- ✓ Sirve como referencia para los sensores del guante, al utilizar el acelerómetro de la DE0-nano.

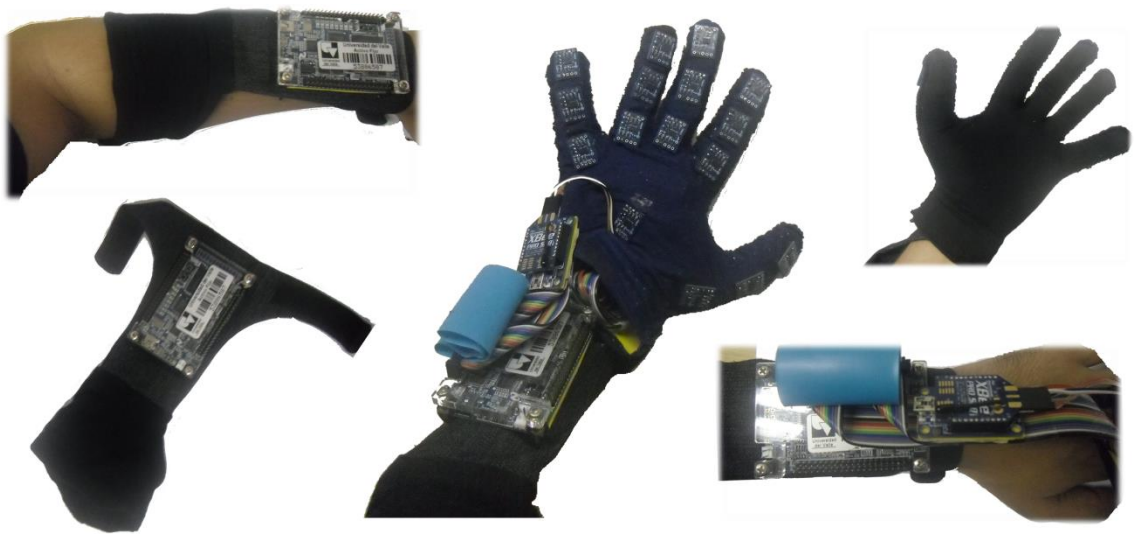


Figura 4.12 Dataglove terminado.

4.2.4. Costos

En la Tabla 4.1 se presenta el costo de los componentes del guante electrónico, considerando solo los materiales. Se observa que el costo total de los componentes es 1'014.000 pesos.

	Cantidad	Costo/unidad	Total
Acelerómetro ADXL345	16	20.000	320.000
DE0-nano	1	350.000	350.000
XBee PRO	2	102.000	204.000
Base Xbee	2	30.000	60.000
Cableado	--	10.000	10.000
Insumos Textiles	--	30.000	30.000
Costo Total	**--**	**--**	**974.000**

Tabla 4.1 Costo de desarrollo del guante

Para obtener el costo del desarrollo total del guante es necesario estimar el valor exacto de la ingeniería no recurrente (NRE) del prototipo desarrollado: diseño del sistema, implementación del diseño y el ensamble manual realizado. Un valor aproximado del guante estimando una producción de 100 unidades, es de $1'500.000 COP (aproximadamente $500 dólares).

4.3. Procesos de Verificación

Unos de los procesos más importantes en el desarrollo de prototipos es el de verificación, el cual permite comprobar que el sistema desarrollado cumple con las especificaciones de diseño, además de servir como método sistémico para el análisis y corrección de errores. El proceso de validación comprueba que el desarrollo cumple con la función para la cual fue diseñado. En esta sección se presentan los procesos de verificación desarrollados.

4.3.1. Verificación a nivel eléctrico

La verificación de la capa física garantiza que todos los componentes electrónicos estén eléctricamente bien instalados y funcionan correctamente, lo cual se hizo a través de mediciones eléctricas de continuidad y tensión. Se hicieron estimaciones y mediciones del consumo de potencia del sistema (Tabla 4.2) donde se observa que el consumo de los ADXL345 es despreciable en comparación con el consumo de la DE0-nano y del XBee.

Objeto de estudio	Estimado (mA/mW)	Stand-by (mA/mW)	Adquisición (mA/mW)
XBee PRO S2B	15 (Stand-By) / 132 (Tx)	52.1 / 172	71.8 / 237
ADXL345	0.018 / 0.06	0.0001 / 0.0033	0.019 / 0.063
DE0-nano	37.6 / 124	106.3 / 351	106.3 / 351
Sistema completo	170 / 561	160.8 / 531	180.4 / 595

Tabla 4.2 Mediciones de consumo de potencia en stand-by y en modo de medición y transmisión

4.3.2. Verificación de la comunicación UART

Esta etapa de verificación tiene el objetivo de garantizar que las tramas enviadas por el módulo UART son recibidas por el dispositivo externo. Este tipo de verificación está orientado a garantizar la integridad de los datos e identificar la máxima tasa de muestreo que se puede utilizar entre el PC y la FPGA sin saturar el canal de comunicación. Se encontró que el XBee tiene un throughput máximo de 35Kbps [3], convirtiéndolo en el cuello de botella del sistema. El throughput máximo de la interfaz UART es 115.2Kbps y el de la interfaz multi I/O SPI implementada es 17Mbps (1Mbps por cada puerto SDIO).

Para el cálculo de ángulos externo se utilizaron tramas de 108 bytes. Un muestreo de 50Hz requiere un ancho de banda de 43.2Kbps, pero el throughput del XBee es 35Kbps, lo que

genera perdida de información y bloqueo de la interfaz. Para un muestreo de 25Hz es necesario un ancho de banda de 21.6Kbps, asi que los dispositivos XBee no se saturan. Cuando se calculan los ángulos de forma local, se utilizan tramas de 24 bytes, alcanzando tasas de muestreo de 100Hz (19.2Kbps).

4.3.3. Verificación a nivel lógico

La DE0-nano incluye un acelerómetro ADXL345 idéntico a los utilizados en el guante, por lo que fue posible implementar y probar los módulos asociados a la interfaz multi I/O SPI implementados en la FPGA, antes de construir la plataforma de adquisición de datos basada en acelerómetros. Esto permitió aislar completamente la presencia de errores lógicos y errores debidos a la instalación eléctrica.

Al acoplar la plataforma de centralización de datos (guante con acelerómetros) con el sistema de procesamiento embebido (DE0-nano), es necesario verificar en los pines de datos (SDIO) la presencia de señales asociadas a los datos generados por los acelerómetros. Si esto no ocurre, lo más probable es que se deba a problemas eléctricos como cables defectuosos, malas conexiones o sensores quemados. También se verificó que el sistema de centralización de datos permite adquirir las mediciones de los acelerómetros a la máxima tasa de muestreo soportada por estos dispositivos (3200Hz) a pesar de que se configuró a la máxima tasa de muestreo del sistema en general (100Hz), limitada por el máximo throughput de los dispositivos XBee.

4.3.4. Verificación a nivel de datos

Esta etapa de verificación consiste en analizar la coherencia y la tendencia de los datos. Mathematica permite visualizar los datos mediante diferentes representaciones visuales, y de forma dinámica (RealTime). Una adecuada representación visual facilita el análisis de la información, haciendo más eficiente la identificación y solución de problemas.

Los datos de los acelerómetros centralizados en la FPGA están en formato paralelo. Considerando las especificaciones del fabricante y la configuración aplicada a los acelerómetros, la sensibilidad debe ser de 3.9mg/LSb, lo que implica una medición de 256 para la magnitud de la gravedad. Se identificó la presencia de offset en las mediciones generadas por los acelerómetros. Este error se puede compensar ajustando el valor de offset en los registros de configuración del acelerómetro, en la FPGA o en Mathematica (Para cálculo de ángulos de forma externa).

En las mediciones, Se observó la presencia de ruido en ciertos bits. A través del análisis de cada uno de los sensores y de conjuntos de sensores, se identificó tres acelerómetros y 2 líneas de conexión defectuosas. Al reemplazar los elementos defectuosos, se solucionó este problema de ruido.

4.3.5. Verificación de la medición de ángulos

En esta prueba se verifican los algoritmos (PC) o hardware configurado (FPGA) que calculan los ángulos presentes en las articulaciones a partir de las mediciones de inclinación adquiridas por los acelerómetros. El proceso consiste en verificar que los ángulos calculados corresponden a los ángulos formados en el guante por cada una de las parejas de acelerómetros asociados.

4.3.6. Comparación con plataformas comerciales similares

En la Tabla 4.3 se compara el sistema desarrollado con algunas plataformas comerciales. Se puede observar que las tres plataformas con mayor exactitud para calcular la postura de la mano, cuestan más de us$5.000, mientras que el guante desarrollado cuesta us$500.

Los guantes de Cyberglove incluyen 22 sensores de flexión ópticos, los cuales permiten calcular los ángulos en todas articulaciones de la mano. El 5DT glove incluye 14 sensores de flexión resistivos, los cuales permiten medir los ángulos de las articulaciones interfalángicas proximales (1 dof) y metacarpofalangicas (2 dof). El DG5 incluye 5 sensores Flex resistivos y mide el patrón de flexión del conjunto de articulaciones que componen cada dedo (no mide los ángulos de las articulaciones). El guante propuesto utiliza 17 acelerómetros y permite calcular los ángulos de la mano asociados a movimientos de flexión y extensión (1 dof) de todas las articulaciones, incluida la muñeca, y funciona muy bien con ejes de rotación horizontales e inclinaciones del eje de rotación de hasta 60°.

Los guantes comerciales comparados tienen una resolución menor a 1°. El sistema propuesto tiene una resolución de 1°, la cual es suficiente para el problema de determinar la postura de la mano. En cuanto a la tasa de muestreo, el sistema propuesto está limitado a 100Hz por la tecnología wireless utilizada (XBee) pero podría superar los 400Hz con tecnologías como bluetooth y wi-fi, sin que el precio cambie significativamente.

	Cyberglove III	Cyberglove II	5DT glove 14	DG5	Propuesto
Precio	>us$30.000	us$17.800	us$5.500	us$585	us$500
Tipo Sensor	Flex óptico, IMU	Flex óptico, IMU	Flex Resistivo, IMU	Flex Resistivo, IMU	Acelerometro MEMS
No. Sensores	22, 1	22, 1	14, 1	5, 1	17
Resolución	<1°	<1°	<1°	<1°	1°
Tasa de muestreo	>100Hz	>100Hz	75Hz	100Hz	100Hz
Conectividad	Wi-fi, usb, SD	Wi-fi, usb	Bluetooth	Wi-fi	XBee

Tabla 4.3. Comparación con guantes electrónicos comerciales [7][8][9][10]

4.4. Referencias

[1] "ADXL345 Datasheet." [Online]. Available: http://www.analog.com/media/en/technical-documentation/data-sheets/ADXL345.pdf. [Accessed: 11-Jun-2015].

[2] "¿Qué es XBee? ." [Online]. Available: http://xbee.cl/que-es-xbee/. [Accessed: 12-Jun-2015].

[3] "ZigBee RF Modules User Guide." [Online]. Available: http://ftp1.digi.com/support/documentation/90000976_W.pdf. [Accessed: 12-Jun-2015].

[4] "DE0-User Manual." [Online]. Available: https://www.altera.com/en_US/pdfs/literature/ug/DE0_Nano_User_Manual_v1.9.pdf. [Accessed: 12-Jun-2015].

[5] "Cyclone IV Device Handbook Volume 3." [Online]. Available: https://www.altera.com/en_US/pdfs/literature/hb/cyclone-iv/cyiv-5v3.pdf. [Accessed: 17-Jun-2015].

[6] "ArduIMU Gloves Pattern." [Online]. Available: http://theglovesproject.com/wp-content/uploads/2013/02/ArduIMUGlovesPatternPDF.pdf. [Accessed: 12-Jun-2015].

[7] "cyberglovesystems.com." [Online]. Available: http://www.cyberglovesystems.com/all-products. [Accessed: 17-Jun-2015].

[8] "Cyberglove II 22 sensor | WorldViz Store." [Online]. Available: http://shop.worldviz.com/shop/cyberglove-ii-22-sensor/. [Accessed: 17-Jun-2015].

[9] "Data Gloves - Virtual Realities." [Online]. Available: https://www.vrealities.com/products/data-gloves. [Accessed: 17-Jun-2015].

[10] "5DT Products." [Online]. Available: http://www.5dt.com/products/pdataglove14.html. [Accessed: 17-Jun-2015].

Capítulo 5
Conclusiones

Los dispositivos MEMS, familia de dispositivos en los que está basado el sistema de medición del sistema electrónico vestible desarrollado, es una tecnología clave para IoT por su diseño compacto y liviano, bajo consumo de potencia y alto desempeño.

El uso exclusivo de acelerómetros MEMS posibilita relacionar las mediciones de aceleración del campo gravitacional con los vectores normales a la superficie dorsal de los segmentos de la mano para calcular los ángulos presentes en las articulaciones, permitiendo determinar la postura de la mano con algunas restricciones.

El cálculo de los ángulos presentes en las articulaciones, a partir de solo mediciones de aceleración del campo gravitacional, presenta excelentes resultados para rotaciones sobre ejes de rotación horizontales con desviaciones de inclinación del eje de rotación de hasta 60°.

La alta configurabilidad que permite la arquitectura de las FPGAs posibilitó implementar: la interfaz multi I/O SPI de 17 puertos y el sistema de cálculo de ángulos concurrente. Implementaciones que serían más complejas de desarrollar sobre dispositivos con arquitecturas menos flexibles.

El cálculo de los ángulos en un dispositivo externo requiere el envío de las mediciones de todos los acelerómetros, para lo cual se utilizan tramas de mayor longitud (108 bytes). Debido al throughtput del XBee, se limita la tasa de muestreo a 25Hz en la comunicación PC-FPGA.

La implementación en hardware (FPGA) del algoritmo para el cálculo de los ángulos incrementó la autonomía de la plataforma de medición, redujo notablemente la carga de procesamiento del dispositivo externo y permitió el uso de tramas más pequeñas (24 bytes), permitiendo alcanzar tasas de muestreo más altas (100Hz) y mejorar el desempeño de todo el sistema.

El algoritmo propuesto para la implementación de la función arco-tangente permite calcular el ángulo considerando sólo el segmento de curva de 0° a 45°, el cual se puede modelar con muy buena precisión con un polinomio de grado 3, por lo que es una alternativa con mucho potencial para implementaciones en sistemas embebidos.

El potencial de Wolfram Mathematica para la adquisición y procesamiento de altos volúmenes de datos, procesar los datos en tiempos muy bajos, y para representar la información de forma gráfica y dinámica, posibilita la implementación de sistemas de verificación y validación dinámicos, visuales y en tiempo real, evitando el uso de metodologías de análisis offline, basadas en largas listas o tablas de datos previamente almacenadas.

Capítulo 6
Recomendaciones para trabajos futuros

En este proyecto se desarrolló un prototipo de tecnología vestible (Dataglove) que utiliza un tipo de cableado básico, el cual puede ser mejorado en las próximas etapas del desarrollo del producto, utilizando sistemas de conexión más confiables y robustos como PCBs flexibles, tinta conductiva, fibras conductivas, etc.

Se recomienda utilizar una tecnología de comunicación inalámbrica más rápida, ya que el XBee fue el principal cuello de botella de esta plataforma, restringiendo la tasa de muestreo. Se recomienda el uso de wi-fi por las tendencias del internet de las cosas (IoT) e IPv6.

El uso exclusivo de mediciones gravitacionales presenta limitaciones a la plataforma. Un sistema equivalente pero utilizando IMUs con magnetómetros en vez de solo acelerómetros, permite el desarrollo de una plataforma más completa y robusta, a partir de la combinación de las mediciones de los sensores integrados, la cual tendría todo el potencial para ser un producto comercial.

El desempeño de la plataforma es muy bueno para las condiciones de operación definidas, y existen muchas aplicaciones y proyectos donde puede ser conveniente el uso de esta plataforma.

Para la aplicación del reconocimiento de la postura de la mano es suficiente una resolución de 1°. Para aplicaciones donde se requiera una mayor resolución en el cálculo de los ángulos, es suficiente con modificar la implementación de la función arco-tangente.

Es importante utilizar una adecuada representación de la información, debido a que en muchas ocasiones es más ágil e importante el uso de criterios cualitativos (aproximaciones, tendencias, comportamientos) que cuantitativos para el análisis de la información.

Wolfram Mathematica es una herramienta computacional muy potente, fácil de usar y con muchos campos de aplicación. Sin importar el área de estudio en que se trabaje, se puede encontrar en ella elementos de mucha utilidad: documentos computables, conectividad con Wólfram Alpha, computación paralela y en la nube, alta conectividad en hardware (dispositivos) y software (Lenguajes de programación, bases de datos, aplicaciones), etc. Finalmente, la Universidad del valle tiene un contrato de licencia con Wolfram en el que todos los funcionarios, estudiantes y computadores inventariados, tienen derecho a una licencia.