El consumo de recursos y la precisión en dispositivos programables: caso de matrices mal condicionadas.

Claudio Guadalupe Cruz Mendoza[1], Dra. Monserrat Morin Castillo[1] Dr. Jose Rubén Conde Sánchez[2], Dr. Jacobo Oliveros Oliveros[2].

[1] Facultad de Ciencias de la Electrónica, de la BUAP, [2] Facultad de Ciencias Físico-Matemáticas de la BUAP.

https://meet.google.com/aht-wnwh-ara

Resumen

En diferentes áreas de las matemáticas y de la instrumentación electrónica existen campos específicos de investigación sobre el uso de las matrices denominadas mal condicionadas. Estas aparecen en diferentes áreas, como son la matemática y la instrumentación. A manera de ejemplo, se puede mencionar el problema de detección de fuentes bioeléctricas en el corazón y en el cerebro[1]. Los sistemas de ecuaciones lineales algebraicas con matrices mal condicionadas están relacionados con la inestabilidad numérica, la cual puede provocar cambios significativos en la solución debido a pequeños cambios en el lado derecho del mencionado sistema.[2] Por otro lado, en el desarrollo de sistemas embebidos es cada vez más común la utilización de hardware especializado para realizar tareas específicas, y una de estas tendencias es el uso de System on Chip (SoC)[3] basados en procesadores trabajando en conjunto con una sección de lógica programable, lo cual proporciona una alternativa extensible y flexible. Por este motivo, se desarrolló una arquitectura de alto desempeño con la que se calcula la inversa de una matriz de 3x3. Para la creación de la arquitectura, se utilizó la Síntesis de Alto Nivel, la cual permite obtener componentes que pueden ser implementados en lógica programable a partir de una función en Lenguaje C. También se utilizó el framework PYNQ el cual permite utilizar el lenguaje Python para la programación de diseños que utilicen microprocesadores y lógica programable. Los resultados de la arquitectura son mostrados a través de su correspondiente funcionamiento con ejemplos que muestra el efecto del mal condicionamiento en la precisión al calcular la inversa de matrices así como en términos de recursos lógicos requeridos.

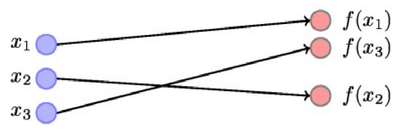
Número de Condición

Mide cuánto se modifica el valor de salida si se realiza un pequeño cambio en el valor de entrada[4].

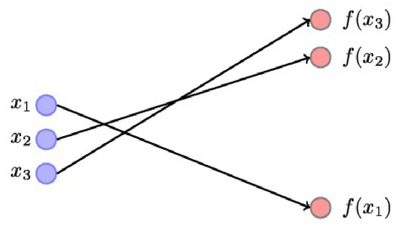
$$k = cond(A) = |||A|||||A^{-1}||| , \quad (1)$$

donde $A \in R^{n\times n}$ y k es el número de condición.

- Si k es pequeño, la matriz está bien condicionada.
- Si k es grande, la matriz está mal condicionada.



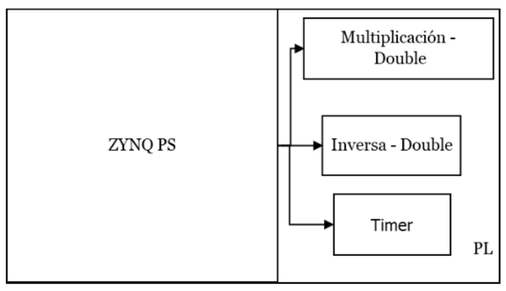
(a) Estabilidad (k es pequeño).



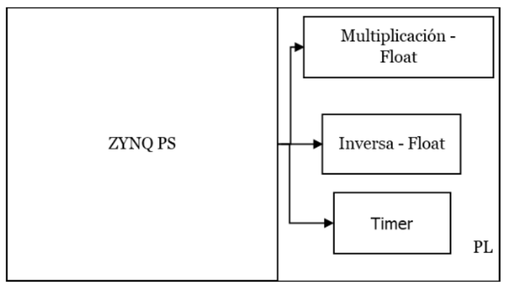
(b) Inestabilidad (k es grande).

Figura 1: a) Pequeños cambios pueden producir pequeños cambios. b) Pequeños cambios pueden producir variaciones sustanciales en la solución buscada.

Arquitectura propuesta



(a) Sistema Double.

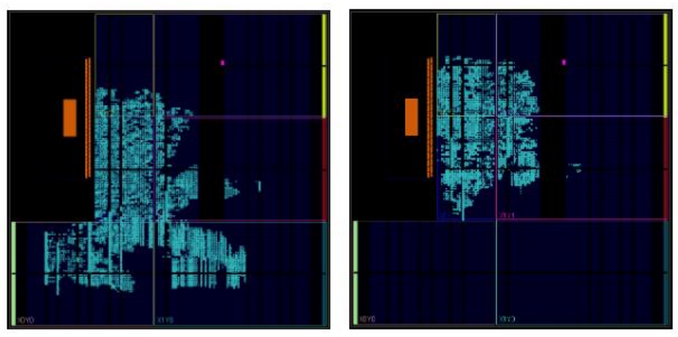


(b) Sistema Float.

Figura 2: Diseños realizados.

Consumo de recursos

En la figura 3 se muestra el área que ocupan los diseños realizados en el SoC.

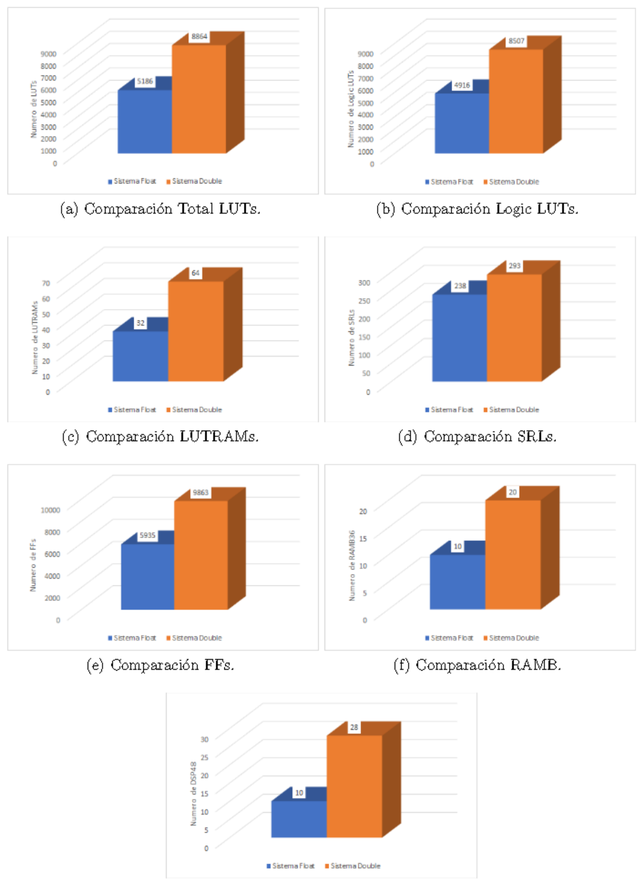


(a) Sistema Double. (b) Sistema Float.

Figura 3. Diseños implementados en la lógica programable de SoC

Tabla 1 Factor de aumento de recursos del Sistema Double respecto al Sistema Float.

Métrica	Factor de crecimiento
Total LUTs	1.7
LOGIC LUTs	1.7
LUTRAMs	2.0
SRLs	1.2
FFs	1.7
RAMB36	2.0
DSP48	2.8



(a) Comparación Total LUTs. (b) Comparación Logic LUTs. (c) Comparación LUTRAMs. (d) Comparación SRLs. (e) Comparación FFs. (f) Comparación RAMB. (g) Comparación DSP.

Figura 4. Comparación de recursos utilizados por ambos sistemas.

Resultados numéricos

Para las pruebas numéricas se tomaron en cuenta matrices mal condicionadas, tomando como referencia la siguiente matriz:

$$A = \begin{pmatrix} 2 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & \epsilon \end{pmatrix}$$

En donde ε toma diferentes valores y está asociado al número de condición de la matriz.

Tabla 2: Resultados obtenidos con $\epsilon = 0{,}011$, $cond(A) = 335{,}962$

CaracterísticaDiseño	Sistema Float	Sistema Double
A^{-1}	$\begin{pmatrix} 0{,}5 & -0{,}5 & 0{,}0 \\ 0{,}0 & 1{,}0 & -90{,}90908813476562 \\ 0{,}0 & 0{,}0 & 90{,}90908813476562 \end{pmatrix}$	$\begin{pmatrix} 0{,}5 & -0{,}5 & 0{,}0 \\ 0{,}0 & 1{,}0 & -90{,}90909090909092 \\ 0{,}0 & 0{,}0 & 90{,}90909090909092 \end{pmatrix}$
Determinante	0.02199999988079071	0.022
$t_{e-inversa}$	0.00030404 segundos	0.00029593 segundos
AA^{-1}	$\begin{pmatrix} 1{,}0 & 0{,}0 & 0{,}0 \\ 0{,}0 & 1{,}0 & 0{,}0 \\ 0{,}0 & 0{,}0 & 0{,}9999999403953552 \end{pmatrix}$	$\begin{pmatrix} 1{,}0 & 0{,}0 & 0{,}0 \\ 0{,}0 & 1{,}0 & 0{,}0 \\ 0{,}0 & 0{,}0 & 1{,}0 \end{pmatrix}$
$t_{e-multiplciacion}$	0.00028458 segundos	0.00028072 segundos

Tabla 3: Resultados obtenidos con $\epsilon = 0{,}000101$

	Sistema Float	Sistema Double
$A^{-}1$	$\begin{pmatrix} 0{,}5 & -0{,}5 & 0{,}0 \\ 0{,}0 & 1{,}0 & -9900{,}990234375 \\ 0{,}0 & 0{,}0 & 9900{,}990234375 \end{pmatrix}$	$\begin{pmatrix} 0{,}5 & -0{,}5 & 0{,}0 \\ 0{,}0 & 1{,}0 & -9900{,}990099009901 \\ 0{,}0 & 0{,}0 & 9900{,}990099009901 \end{pmatrix}$
Determinante	0.00020199999562464654	0.000202
$t_e - inversa$	0.00029604 segundos	0.00029286 segundos
$AA^{-}1$	$\begin{pmatrix} 1{,}0 & 0{,}0 & 0{,}0 \\ 0{,}0 & 1{,}0 & 0{,}0 \\ 0{,}0 & 0{,}0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1{,}0 & 0{,}0 & 0{,}0 \\ 0{,}0 & 1{,}0 & 0{,}0 \\ 0{,}0 & 0{,}0 & 1{,}0 \end{pmatrix}$
$t_e - multiplciacion$	0.00028108 segundos	0.00028236 segundos

La prueba con redondeo se hace, similarmente, sobre un diseño a la vez (Sistema Float o Sistema Double), sin embargo, las operaciones se hacen sobre una matriz con un ε el cual toma una cantidad limitada de números decimales.

Tabla 4: Comparación entre diferente cantidad de dígitos de truncamiento utilizando el "Sistema Float".

Dígitos de Truncamiento	8	16	32
ϵ_{aprox}	$5{,}99999978589949 \times 10^{-8}$	$6{,}666666507726404 \times 10^{-8}$	$6{,}666666507726404 \times 10^{-8}$
$cond(A)$	6×10^{7}	6×10^{7}	6×10^{7}
$cond(A_{aprox})$	$6{,}1592 \times 10^{7}$	$5{,}5433 \times 10^{7}$	$5{,}5433 \times 10^{7}$
A^{-1}_{aprox}	$\begin{pmatrix} 0{,}5 & -0{,}5 & 0{,}0 \\ 0{,}0 & 1{,}0 & -16666667{,}0 \\ 0{,}0 & 0{,}0 & 16666667{,}0 \end{pmatrix}$	$\begin{pmatrix} 0{,}5 & -0{,}5 & 0{,}0 \\ 0{,}0 & 1{,}0 & -15000000{,}0 \\ 0{,}0 & 0{,}0 & 15000000{,}0 \end{pmatrix}$	$\begin{pmatrix} 0{,}5 & -0{,}5 & 0{,}0 \\ 0{,}0 & 1{,}0 & -15000000{,}0 \\ 0{,}0 & 0{,}0 & 15000000{,}0 \end{pmatrix}$
$det(A_{aprox})$	1.199999957179898e-07	1.3333333015452808e-07	1.3333333015452808e-07

Tabla 5: Comparación entre diferente cantidad de dígitos de truncamiento utilizando el "Sistema Double".

Dígitos de Truncamiento	8	16	32
ϵ_{aprox}	6×10^{-8}	$6{,}66666666 \times 10^{-8}$	$6{,}666666666666667 \times 10^{-8}$
$cond(A)$	6×10^{7}	6×10^{7}	6×10^{7}
$cond(A_{aprox})$	$6{,}1592 \times 10^{7}$	$5{,}5433 \times 10^{7}$	$5{,}5433 \times 10^{7}$
A^{-1}_{aprox}	$\begin{pmatrix} 0{,}5 & -0{,}5 & 0{,}0 \\ 0{,}0 & 1{,}0 & -16666666{,}666666668 \\ 0{,}0 & 0{,}0 & 16666666{,}666666668 \end{pmatrix}$	$\begin{pmatrix} 0{,}5 & -0{,}5 & 0{,}0 \\ 0{,}0 & 1{,}0 & -15000000{,}015 \\ 0{,}0 & 0{,}0 & 15000000{,}015 \end{pmatrix}$	$\begin{pmatrix} 0{,}5 & -0{,}5 & 0{,}0 \\ 0{,}0 & 1{,}0 & -15000000{,}0 \\ 0{,}0 & 0{,}0 & 15000000{,}0 \end{pmatrix}$
$det(A_{aprox})$	$1{,}2 \times 10^{-7}$	$1{,}333333332 \times 10^{-7}$	$1{,}3333333333333334 \times 10^{-7}$

Conclusiones: Con matrices mal condicionadas, se requiere un mayor consumo de recursos para obtener la precisión requerida. En el cálculo de la inversa existe una diferencia sustancial entre ambos tipos de datos, donde el Sistema Double arrojó resultados mejores, sin embargo la cantidad de recursos se disparó al doble en muchos elementos. Se puede apreciar que al redondear a 16 números decimales se obtienen resultados aceptables para ambos casos, por lo que el Sistema Float, tiene un desempeño mejor a comparación del Sistema Double, sobre todo tomando en cuenta los recursos utilizados por ambos Sistemas.

Referencias:

1) An FPGA-based analysis of trade-offs in the presence of ill-conditioning and different precision levels in computations Algredo-Badillo I, Conde-Mones JJ, Hernández-Gracidas CA, Morín-Castillo MM, Oliveros-Oliveros JJ, et al. (2020)
2) J. Rey Cabezas J. Infante del Río. Métodos Numéricos, Teoría, problemas y prácticas con MATLAB. 4ta Edición. Ciencia y Técnica. Pirámide, 2002
3) Louise H. Crockett y col. The Zynq Book: Embedded Processing with the Arm Cortex-A9 on the Xilinx Zynq-7000 All Programmable Soc. 1st Edition. Strathclyde Academic Media, 2014.
4) Edward Rothwell y B. Drachman. "Unified approach to solving ill-conditioned matrix problems". En: International Journal for Numerical Methods in Engineering 28 (mar. de 1989), págs. 609-620.