

Rapport de Projet

5^e année du cycle ingénieur

Spécialité : Électronique et Informatique pour l'Embarqué

Polytech Paris-Saclay

<p>Designing Generic Square Root Architectures on FPGAs Using VHDL</p>

Étudiants : Fatima ISMAIL, Ali MAKKI

Présenté à : Mathias Hervé

Projet : FPGA

Decembre 2025

L'objectif de ce projet est de concevoir sur FPGA différentes architectures mettant en œuvre trois algorithmes de calcul de racine carrée, chacun répondant à des contraintes de conception différentes. Ces architectures seront conçues et testées en VHDL. Les circuits calculeront la racine carrée d'une entrée non signée A , de taille $2n$ bits.

1 Software Implementation

1.1 Algorithm 1 : Newton's method

C Implementation

```
unsigned int sqrt1(unsigned int Xio){
if (Xio == 0) return 0;
unsigned int x = Xio;
unsigned int x_prev;
do {
x_prev = x;
x = (x + Xio / x) » 1;
} while (x < x_prev);
return x;
}
```

We obtain, for $Xio = 2,550,409$:

- res = 1597
- Duration of `sqrt1` = 2383

1.2 Algorithm 2 : Iterative square root algorithm based on successive approximations

C Implementation

```
unsigned int sqrt2(unsigned int Xio){
unsigned int X = 1U « 30;
unsigned int V = 1U « 14;
unsigned int Z = 1U « 15;
int i;
for (i=14; i>=0; i--) {
if (X>Xio) {
X = X + V*(V-(Z«1));
Z = Z - V;}
else if (X<Xio) {
X = X + V*(V+(Z«1));
Z = Z + V;}
V = (V»1);
}
if (X>Xio){
return Z-1;
} else {
return Z;}
}
```

For $Xio = 2,550,409$:

- res = 1597
- Duration of `sqrt1` = 976

1.3 Algorithm 3 : Iterative square root algorithm by reasoning on successive remainders

C Implementation

```
unsigned int sqrt3(unsigned int Xio){
    unsigned int V =1«30;
    unsigned int Z=0;
    int i;
    for (i=15; i>=0; i-) {
        Z= Z+V;
        if (Xio>=Z) {
            Xio=Xio-Z;
            Z=Z+V;}
        else{
            Z=Z-V;}
        Z=Z»1;
        V=V»2;}
    return Z;
}
```

For $Xio = 2,550,409$:

- res = 1597
- Duration of `sqrt1` = 903

Conclusion

Les trois algorithmes produisent le même résultat correct, mais leurs temps d'exécution diffèrent nettement. La méthode de Newton est la plus lente (2383 unités) en raison des divisions répétées. La méthode d'approximation successive est plus rapide (976 ms), et l'algorithme basé sur le reste offre les meilleures performances (903 ms). Ce dernier allie rapidité et simplicité, ce qui en fait l'option la plus adaptée pour une implémentation sur FPGA.

2 Hardware Implementation

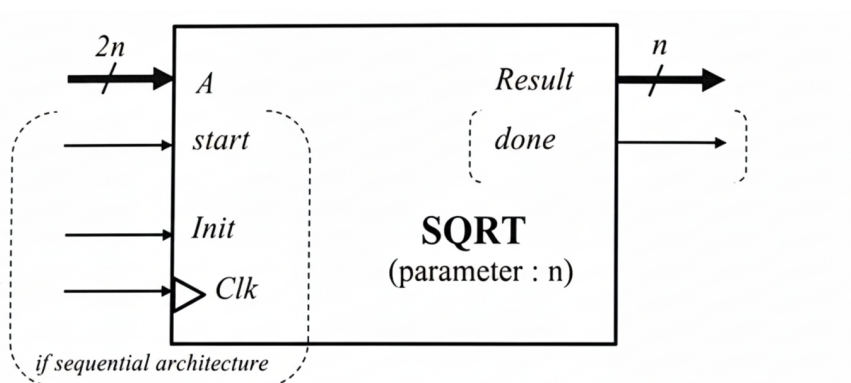


FIGURE 1 – Top-level view of the sqrt circuit

2.1 Behavioural approach

Toutes les architectures sont conçues en VHDL en s'appuyant sur un seul processus.

VHDL Code – Sequential Square Root Entity

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity sqrt_seq is
generic(N : NATURAL := 64);
port(
A : in std_logic_vector(N-1 downto 0);
start : in std_logic;
init : in std_logic;
clk : in std_logic;
result : out std_logic_vector(N/2 -1 downto 0);
done : out std_logic
);
end sqrt_seq;
```

2.2 Architecture 1

L'architecture séquentielle est conçue pour implémenter la méthode de Newton tout en minimisant le nombre d'états nécessaires au calcul. Chaque itération de l'algorithme est exécutée en un seul cycle d'horloge.

Cette architecture est testée à l'aide de testbenches écrits en VHDL, Cette image met en évidence la plus longue séquence de cycles d'horloge 23 au total située entre les deux lignes verticales jaunes :

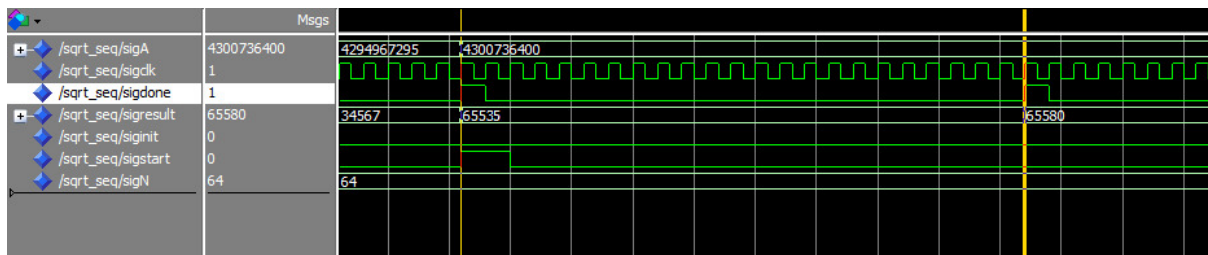


FIGURE 2 – Architecture 1 in testbench

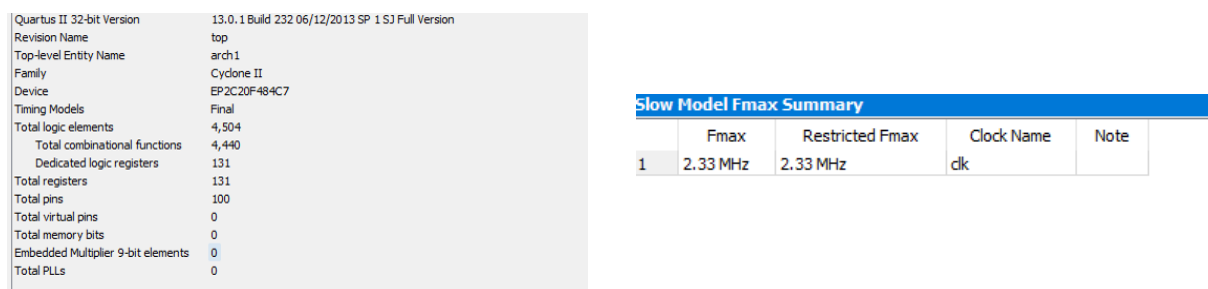


FIGURE 4 – Fmax arch1

FIGURE 3 – Synthese

Analyse de l'Architecture 1

Principe de fonctionnement L'architecture 1 implémente la méthode de Newton-Raphson pour le calcul de la racine carrée. Cette méthode itérative converge rapidement vers la solution en utilisant la formule de récurrence :

$$x_{n+1} = \frac{1}{2} \left(x_n + \frac{A}{x_n} \right)$$

Machine à états La machine à états comporte trois états principaux :

- **État s0 (Idle)** : Attente du signal **start**. Initialisation de x avec la valeur d'entrée A .
- **État s1 (Calcul)** : Exécution de l'itération de Newton. À chaque cycle, une nouvelle approximation est calculée. La convergence est détectée lorsque $|x - x_{prev}| \leq 1$.
- **État s2 (Terminé)** : Signal **done** activé, résultat disponible sur la sortie **result**.

Avantages et inconvénients

Avantages :

- Algorithme bien connu et facile à comprendre

Inconvénients :

- Utilisation intensive de ressources (4530 unités logiques)
- Division coûteuse en matériel (opération non native sur FPGA)
- Fréquence maximale très faible (2,33 MHz) due au chemin critique de la division
- Temps d'exécution total élevé malgré peu d'itérations

2.3 Architecture 2

Cette architecture est séquentielle et conçue pour implémenter le deuxième algorithme tout en minimisant le nombre d'états nécessaires au calcul. Chaque itération du processus est exécutée en un seul cycle d'horloge.

Cette architecture est testée à l'aide de testbenches écrits en VHDL :

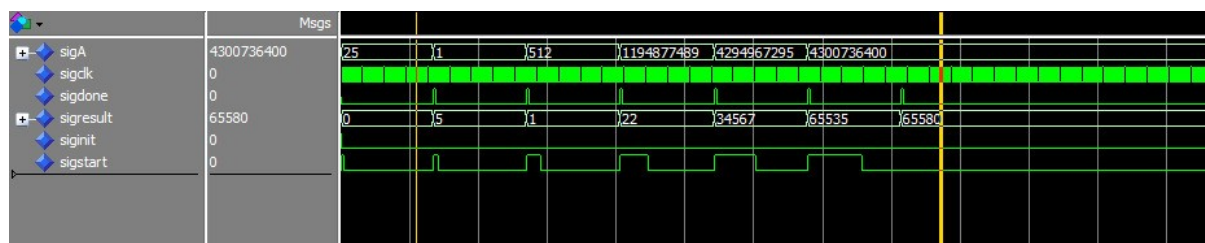


FIGURE 5 – Architecture 2 in testbench

Quartus II 32-bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Full Version
Revision Name	top
Top-level Entity Name	arch2
Family	Cyclone II
Device	EP2C20F484C7
Timing Models	Final
Total logic elements	715 / 18,752 (4 %)
Total combinational functions	670 / 18,752 (4 %)
Dedicated logic registers	234 / 18,752 (1 %)
Total registers	234
Total pins	100 / 315 (32 %)
Total virtual pins	0
Total memory bits	0 / 239,616 (0 %)
Embedded Multiplier 9-bit elements	22 / 52 (42 %)
Total PLLs	0 / 4 (0 %)

Slow Model Fmax Summary				
	Fmax	Restricted Fmax	Clock Name	Note
1	51.43 MHz	51.43 MHz	clk	

FIGURE 7 – Fmax arch2

FIGURE 6 – Synthese arch2

Analyse de l'Architecture 2

Principe de fonctionnement L'architecture 2 utilise un algorithme d'approximations successives qui évite la division. L'algorithme maintient trois variables :

- X : approximation du carré de la racine recherchée
- Z : approximation de la racine carrée
- V : pas de variation, divisé par 2 à chaque itération

À chaque itération, l'algorithme compare X à A et ajuste Z en conséquence en ajoutant ou retirant V .

Détail des opérations À chaque cycle d'horloge :

1. Comparaison de X avec A
2. Si $X > A$: mise à jour $X \leftarrow X + V(V - 2Z)$ et $Z \leftarrow Z - V$
3. Si $X < A$: mise à jour $X \leftarrow X + V(V + 2Z)$ et $Z \leftarrow Z + V$
4. Division de V par 2 (décalage à droite d'un bit)
5. Décrémentement du compteur d'itérations

Avantages et inconvénients

Avantages :

- Élimination de la division : uniquement multiplications et additions
- Nombre d'itérations fixe et prévisible ($N/2$ itérations pour N bits)
- Fréquence maximale nettement améliorée (51,43 MHz)
- Réduction drastique des ressources (715 vs 4530)
- Temps d'exécution total réduit d'un facteur 15

Inconvénients :

- Plus d'itérations nécessaires (34 cycles contre 23)
- Utilisation de multiplications qui restent coûteuses
- Latence fixe même pour des valeurs simples

2.4 Architecture 3

Cette architecture est séquentielle et conçue pour implémenter le troisième algorithme tout en minimisant le nombre d'états nécessaires au calcul. Chaque itération du processus est exécutée en un seul cycle d'horloge.

Cette architecture est testée à l'aide de testbenches écrits en VHDL :

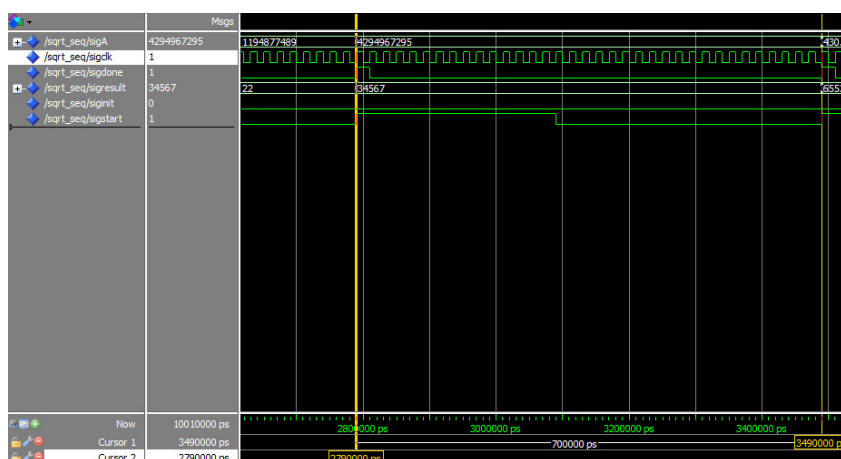


FIGURE 8 – Architecture 3 in testbench

Flow Status:	Successful - Fri Nov 28 09:05:36 2025
Quartus II 32-bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Full Version
Revision Name	top
Top-level Entity Name	arch3
Family	Cyclone II
Device	EP2C20F484C7
Timing Models	Final
Total logic elements	505 / 18,752 (3 %)
Total combinational functions	437 / 18,752 (2 %)
Dedicated logic registers	202 / 18,752 (1 %)
Total registers	202
Total pins	100 / 315 (32 %)
Total virtual pins	0
Total memory bits	0 / 239,616 (0 %)
Embedded Multiplier 9-bit elements	0 / 52 (0 %)
Total PLLs	0 / 4 (0 %)

Slow Model Fmax Summary				
	Fmax	Restricted Fmax	Clock Name	Note
1	83.1 MHz	83.1 MHz	clk	

FIGURE 10 – Fmax arch3

FIGURE 9 – Synthese arch3

Analyse de l'Architecture 3

Algorithme par restes successifs L'architecture 3 implémente l'algorithme 3, basé sur le calcul de la racine carrée par raisonnement sur les restes successifs. Cet algorithme évite complètement les multiplications en utilisant uniquement des additions, soustractions et décalages.

Principe de fonctionnement L'algorithme construit la racine carrée bit par bit, de gauche à droite (du bit de poids fort au bit de poids faible). À chaque itération :

1. On teste si l'ajout d'un nouveau bit à 1 dans le résultat Z donnerait un carré inférieur ou égal au reste courant
2. Si oui, on garde ce bit à 1 et on soustrait la valeur correspondante du reste
3. Sinon, on met ce bit à 0
4. On décale les variables pour traiter le bit suivant

Variables utilisées

- V : valeur de test, initialisée à $2^{2(N/2-1)}$ et divisée par 4 à chaque itération
- Z : résultat accumulé de la racine carrée
- Xio : reste courant, initialement égal à l'entrée A

Détail des opérations par cycle À chaque cycle d'horloge :

```

Z = Z + V                // Test du bit suivant
if (Xio >= Z) then
    Xio = Xio - Z         // Le bit est valide
    Z = Z + V             // On le garde
else
    Z = Z - V             // Le bit n'est pas valide
end if
Z = Z >> 1               // Décalage à droite
V = V >> 2               // Division par 4

```

Avantages de l'absence de multiplication Cette architecture présente des caractéristiques uniques :

- **Aucun multiplieur** : uniquement des additionneurs/soustracteurs et des décaleurs
- **Opérations simples** : tous les chemins critiques sont courts
- **Économie de ressources** : 505 Unites logiques
- **Haute fréquence** : 83,1 MHz grâce à la simplicité des opérations

Métriques de performance Nombre d'itérations : Fixe

Cycles d'horloge : 35 cycles

Temps d'exécution : 421,17 ns à 83,1 MHz

Ressources : Seulement 505 (le plus efficace)

Avantages et inconvénients

Avantages :

- Fréquence la plus élevée (83,1 MHz) parmi les architectures séquentielles
- Utilisation minimale des ressources (505 unités)
- Temps d'exécution le plus court (421 ns)
- Algorithme déterministe et prédictible
- Simplicité de l'implémentation
- Excellent rapport performance/ressources

Inconvénients :

- Latence fixe même pour des valeurs simples
- Plus d'itérations que la méthode de Newton (N/2 vs 23)
- Débit limité : une seule opération à la fois

Comparaison avec les architectures 1 2 et 3 La comparaison des trois premières architectures révèle des différences de performance significatives.

	archi1	archi2	archi3
Ressources	4530	715	505
Fmax	2.33 MHz	51.43 MHz	83.1 MHz
Nb of clk cycles	23	34	35
Temps (ns)	9871.14	661.093	421.17

L'Architecture 1, basée sur la méthode de Newton, souffre d'une opération de division extrêmement coûteuse qui limite sa fréquence maximale à seulement 2,33 MHz. Malgré un nombre d'itérations réduit (23 cycles), son temps d'exécution atteint 9871 ns, soit 23 fois plus lent que l'Architecture 3. De plus, elle consomme 4530 LUT, presque 9 fois plus de ressources.

L'Architecture 2 élimine la division en utilisant des multiplications, ce qui améliore considérablement la fréquence à 51,43 MHz et réduit le temps d'exécution à 661 ns. Elle nécessite 715. Avec 34 cycles d'horloge, elle offre un compromis raisonnable entre performance et ressources.

L'Architecture 3 se distingue comme la solution optimale en évitant complètement les multiplications et divisions. Utilisant uniquement des additions, soustractions et décalages, elle atteint la fréquence la plus élevée (83,1 MHz) avec le minimum de ressources (505 LUT). Son temps d'exécution de 421 ns en fait l'architecture la plus rapide. Ce résultat démontre qu'en conception FPGA, la simplicité des opérations et la réduction du chemin critique sont plus déterminantes que le nombre d'itérations. L'Architecture 3 représente donc le meilleur choix pour la majorité des applications embarquées nécessitant le calcul de racines carrées.

2.5 Architecture 4 - Combinatoire

L'architecture 4 déploie l'algorithme 3 (par restes successifs) entièrement en logique combinatoire, sans registres intermédiaires. Toutes les 32 itérations sont réalisées dans un seul grand circuit combinatoire, sans multiplication.

Cette architecture est testée à l'aide de testbenches écrits en VHDL :



FIGURE 11 – Architecture 4 in testbench

Flow Summary	
Flow Status:	Successful - Fri Nov 28 12:07:50 2025
Quartus II 32-bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Full Version
Revision Name	top
Top-level Entity Name	arch4
Family	Cyclone II
Device	EP2C20F484C7
Timing Models	Final
Total logic elements	2,675 / 18,752 (14 %)
Total combinational functions	2,655 / 18,752 (14 %)
Dedicated logic registers	96 / 18,752 (< 1 %)
Total registers	96
Total pins	97 / 315 (31 %)
Total virtual pins	0
Total memory bits	0 / 239,616 (0 %)
Embedded Multiplier 9-bit elements	0 / 52 (0 %)
Total PLLs	0 / 4 (0 %)

Slow Model Fmax Summary				
	Fmax	Restricted Fmax	Clock Name	Note
1	6.34 MHz	6.34 MHz	clk	

FIGURE 13 – Fmax acrh4

FIGURE 12 – Synthese arch4

Analyse de l'Architecture 4

Structure du circuit Le circuit est constitué de $N/2$ blocs combinatoires en cascade :

- Chaque bloc représente une itération complète de l'algorithme 3
- Chaque bloc contient : un additionneur, un soustracteur, un comparateur et des décaleurs
- Les sorties d'un bloc (Z , Xio , V) sont directement connectées aux entrées du bloc suivant
- Aucune synchronisation par horloge entre les blocs
- Le signal de sortie est disponible après propagation à travers tous les blocs

Timing et chemin critique Le chemin critique traverse l'ensemble des 16 itérations, créant un délai de propagation de 157 ns. Ce long délai explique la faible fréquence maximale de 6,34 MHz (période minimale = 157 ns). Malgré l'absence de multiplications, la cascade de 16 additionneurs/soustracteurs crée un délai significatif.

Avantages et inconvénients

Avantages :

- Résultat disponible en un seul cycle d'horloge
- Pas de machine à états ni de logique de contrôle
- Latence minimale en nombre de cycles (1 cycle)
- Simplicité conceptuelle

Inconvénients :

- Chemin critique très long (157 ns)
- Fréquence maximale très limitée (6,34 MHz)
- Consommation de ressources importante (2579 unités)
- Consommation énergétique élevée (toute la logique commute à chaque calcul)
- Difficultés de timing closure lors de la synthèse
- Non évolutif pour des largeurs de données importantes

Cas d'usage Cette architecture convient uniquement pour :

- Des applications nécessitant une latence absolue minimale
- Des systèmes où la fréquence d'horloge est intrinsèquement basse
- Des calculs occasionnels où le débit n'est pas critique

2.6 Architecture 5 - Pipelined Architecture

L'architecture 5 représente un compromis entre l'architecture combinatoire pure (arch4) et l'architecture séquentielle (arch3). Elle découpe le circuit combinatoire de l'architecture 4 en plusieurs étages de pipeline, tout en conservant l'algorithme 3 sans multiplications. Cette architecture est testée à l'aide de testbenches écrits en VHDL :

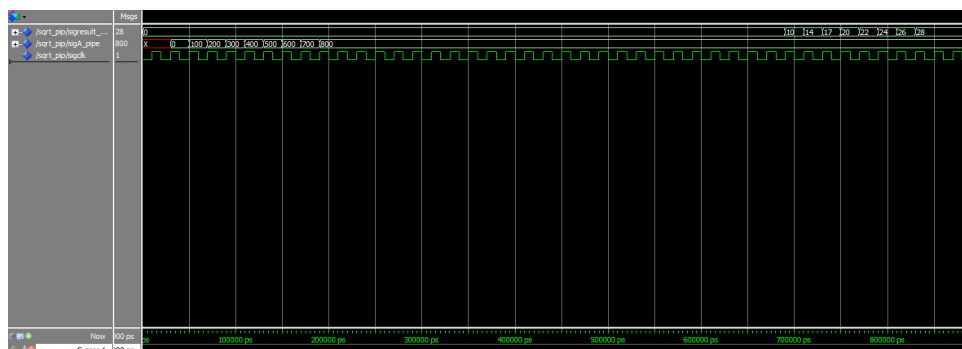


FIGURE 14 – Architecture 5 in testbench

Flow Summary	
Flow Status:	Successful - Fri Nov 28 12:11:37 2025
Quartus II 32-bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Full Version
Revision Name	top
Top-level Entity Name	arch5
Family	Cyclone II
Device	EP2C20F484C7
Timing Models	Final
Total logic elements	2,994 / 18,752 (16 %)
Total combinational functions	1,758 / 18,752 (9 %)
Dedicated logic registers	2,641 / 18,752 (14 %)
Total registers	2641
Total pins	100 / 315 (32 %)
Total virtual pins	0
Total memory bits	0 / 239,616 (0 %)
Embedded Multiplier 9-bit elements	0 / 52 (0 %)
Total PLLs	0 / 4 (0 %)

Slow Model Fmax Summary				
	Fmax	Restricted Fmax	Clock Name	Note
1	126.23 MHz	126.23 MHz	clk	

FIGURE 16 – Fmax arch5

FIGURE 15 – Synthese arch5

Analyse de l'Architecture 5

Stratégie de découpage - Pipeline complet Cette architecture implémente un pipeline complet où chaque itération de l'algorithme 3 constitue un étage distinct :

- **Un étage = une itération** : 32 étages pour $N = 64$ bits (sortie 32 bits)
- Chaque étage contient les registres pour stocker AA , Z et V
- La logique combinatoire d'un étage effectue : une addition, une comparaison, une soustraction conditionnelle, et deux décalages
- Un signal `valid` propage la validité des données à travers le pipeline
- Avantage : chemin critique minimal (une seule itération par cycle), fréquence maximale élevée

Analyse des performances

Amélioration de la fréquence : La fréquence passe de 6,34 MHz (arch4 combinatoire) à 126,23 MHz (arch5), soit une multiplication par 20. Cette amélioration spectaculaire est due au fait que chaque étage du pipeline ne contient qu'une seule itération, réduisant drastiquement le chemin critique.

Latence : 33 cycles contre 1 cycle pour l'architecture combinatoire. Cependant, avec une horloge $20\times$ plus rapide, le temps réel pour le premier résultat reste compétitif (261 ns vs 157 ns).

Débit : Après remplissage du pipeline (33 cycles), un nouveau résultat est produit à chaque cycle d'horloge. À 126,23 MHz, cela représente un débit théorique de 126 millions de racines carrées par seconde, ce qui est idéal pour le traitement continu de flux de données.

Avantages et inconvénients

Avantages :

- Fréquence maximale très élevée (126,23 MHz) - la plus haute de toutes les architectures
- Débit maximal : un résultat par cycle après remplissage du pipeline
- Idéal pour traitement de flux continu (vidéo, audio, DSP)
- Temps d'exécution compétitif (261 ns pour le premier résultat)
- Pipeline complet : exploitation maximale du parallélisme
- Pas de multiplications : utilise seulement additions, soustractions et décalages
- Utilisation efficace des ressources FPGA modernes

Inconvénients :

- Latence initiale élevée (33 cycles avant le premier résultat)
- Ressources plus importantes que l'architecture séquentielle (2994 vs 505) dues aux registres de pipeline
- Inefficace pour des calculs isolés ou sporadiques
- Consommation énergétique plus élevée (tous les étages du pipeline sont actifs)

2.7 Architecture 6 – Sequential Architecture without Multiplier

Flow Summary	
Flow Status	Successful - Mon Dec 01 16:18:23 2025
Quartus II 32-bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Full Version
Revision Name	top
Top-level Entity Name	sqr_t_top
Family	Cyclone II
Device	EP2C20F484C7
Timing Models	Final
Total logic elements	1,028 / 18,752 (5 %)
Total combinational functions	987 / 18,752 (5 %)
Dedicated logic registers	236 / 18,752 (1 %)
Total registers	236
Total pins	100 / 315 (32 %)
Total virtual pins	0
Total memory bits	0 / 239,616 (0 %)
Embedded Multiplier 9-bit elements	0 / 52 (0 %)
Total PLLs	0 / 4 (0 %)

Slow Model Fmax Summary				
	Fmax	Restricted Fmax	Clock Name	Note
1	59.0 MHz	59.0 MHz	clk	

FIGURE 18 – Fmax acrh6

FIGURE 17 – Synthese arch6

Analyse de l'Architecture 6

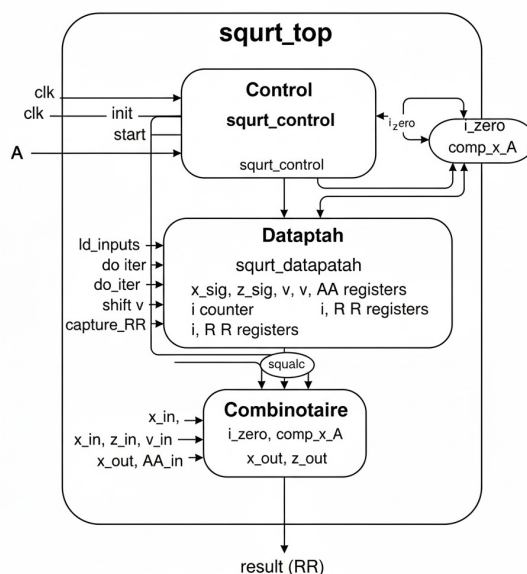


FIGURE 19 – Chemain Block arch6

Algorithme 2 sans multiplieur L'architecture 6 implémente l'algorithme 2 (approximations successives) mais en évitant l'utilisation de multiplieurs matériels. Les multiplications sont remplacées par des séquences d'additions et de décalages. Cette approche réduit significativement l'utilisation des ressources DSP (Digital Signal Processing) du FPGA.

Technique de remplacement Les multiplications sont remplacées par :

- **Décalages** pour les multiplications par puissances de 2
- **Additions successives** pour les autres cas
- **Décomposition** des multiplications complexes en opérations plus simples

Exemple : $V \times (V + 2Z)$ devient une séquence d'opérations :

```
temp1 = shift_left(Z, 1)  -- 2Z par décalage
temp2 = V + temp1
temp3 = 0
for i in 0 to width(V)-1:
    if V(i) = '1' then
        temp3 = temp3 + shift_left(temp2, i)
```

Organisation modulaire de l'architecture 6 Contrairement aux autres architectures présentées dans ce rapport, qui sont intégralement décrites dans un seul fichier VHDL, l'architecture 6 adopte une organisation pleinement modulaire. Elle est décomposée en plusieurs blocs distincts :

- un **datapath** (`sqrt_datapath`) chargé des opérations arithmétiques, du calcul de R , des décalages et des additions successives ;
- une **unité de contrôle** (`sqrt_control`) qui pilote la séquence des opérations et génère les signaux de synchronisation (*ld_inputs*, *shift_v*, *do_iter*, etc.) ;
- un **bloc spécialisé remplaçant la multiplication** (`mul_shift_add`), chargé de substituer les produits matériels par une combinaison de décalages et d'accumulations successives ;
- un **top-level** (`sqrt_top`) qui instancie les différents modules et assure l'interface externe.

Le bloc `mul_shift_add` constitue l'élément essentiel de cette architecture. Il permet de réaliser des expressions du type $V \times (V + 2Z)$ sans aucun multiplieur matériel, seulement au moyen de décalages et d'additions successives. Ce choix réduit drastiquement l'utilisation des DSP du FPGA, au prix d'un allongement du chemin critique dû au nombre accru d'opérations logiques.

Impact sur les performances

Fréquence : 59 MHz, supérieure à l'architecture 2 (51,43 MHz)

Temps total : 576 ns, intermédiaire entre arch2 (661 ns) et arch5 (261 ns)

Avantages de l'organisation modulaire L'introduction de modules séparés apporte plusieurs bénéfices concrets :

- **Lisibilité et maintenance améliorées** : chaque bloc possède une responsabilité bien définie.
- **Réutilisabilité** : le bloc `mul_shift_add` peut être réutilisé dans d'autres conceptions nécessitant des multiplications sans DSP.
- **Facilité d'évolution** : l'unité de contrôle et le datapath peuvent être modifiés indépendamment.
- **Analyse temporelle plus simple** : chaque module peut être vérifié et optimisé séparément.

Inconvénients liés à cette organisation Cette modularité apporte aussi certaines limites :

- **Complexité structurelle accrue** : plusieurs fichiers, plus de signaux internes, une architecture globale plus large.

- **Chemin critique potentiellement plus long** : la suppression du multiplieur matériel impose davantage d'opérations séquentielles dans le bloc `mul_shift_add`.
- **Légère baisse de fréquence** par rapport à une architecture reposant sur des multiplieurs DSP.

Avantages et inconvénients sur les performances

Avantages :

- Réduction drastique des ressources matérielles
- Portable sur FPGAs bas de gamme sans multiplieurs

Inconvénients :

- Fréquence réduite par rapport à l'architecture 5 (59 MHz vs 126 MHz)
- Temps d'exécution plus long que les architectures avec pipeline
- Complexité accrue du code VHDL pour implémenter les multiplications

2.8 Synthèse comparative des architectures

Classification par algorithme

- **Algorithme 1 (Newton)** : Architecture 1 (séquentielle avec division)
- **Algorithme 2 (Approximations successives)** :
 - Architecture 2 (séquentielle avec multiplieurs)
 - Architecture 6 (séquentielle sans multiplieurs)
- **Algorithme 3 (Restes successifs)** :
 - Architecture 3 (séquentielle, pas de multiplication)
 - Architecture 4 (combinatoire, pas de multiplication)
 - Architecture 5 (pipeline , pas de multiplication)

Critère	Arch1	Arch2	Arch3	Arch4	Arch5	Arch6
Algorithme	Newton	Algo 2	Algo 3	Algo 3	Algo 3	Algo 2
Ressources (LUT)	4530	715	505	2579	2994	1028
F_{max} (MHz)	2,33	51,43	83,1	6,34	126,23	59
Cycles d'horloge	23	34	35	1	33	34
Temps total (ns)	9871	661	421	157	261	576
Multiplications	Oui (div)	Oui	Non	Non	Non	Non
Débit	Faible	Moyen	Moyen	Faible	Élevé	Moyen
Latence	Moyenne	Élevée	Élevée	Min	Élevée	Élevée
Type	Séq.	Séq.	Séq.	Comb.	Pipeline	Séq.

TABLE 1 – Comparaison détaillée des six architectures