مجلة جامعة طرطوس للبحوث والدراسات العلمية_ سلسلة العلوم الهندسية المجلد (3) العدد (٥) 2019
Tartous University Journal for Research and Scientific Studies- Engineering Sciences Series Vol. (3) No. (5) 2019

استخدام معالجات الأغراض العامة في النظم الفيزيائية المحوسبة باعتماد ذاكرة SPM

سوزي صالح*
محمد ملحم**
رزان عقباتي***

(تاريخ الإيداع ١٦ / ٦ / ٢٠١٩ . قُبل للنشر ١٢ / ٩ / ٢٠١٩)

الملخص

تُعرّف النظم الفيزيائية المحوسبة :(Cyber Physical Systems) CPS أنها الجيل الجديد من الأنظمة المضمنة، حيث تتكامل فيها نظم الحوسبة والشبكات الحاسوبية والعمليات الفيزيائية. إذ تعمل الحوسبة المضمنة والشبكات على إدارة العمليات الفيزيائية، وتعمل حلقات التغذية العكسية (feedback) على ربط العملية الفيزيائية مع نظام الحوسبة. لذلك يحتاج تطبيق نظم CPS إلى مواءمة وتزامن كبير بين هذه الأنواع الثلاثة المختلفة من الأنظمة.

إن استخدام معالجات الأغراض العامة ببنيتها المعروفة في تصميم أنظمة CPS لن يكون مُجدياً من ناحية ماتقدمه هذه البنية من عدم القدرة على استيعاب المواصفات الزمنية لأنظمة CPS. يتضمن هذا البحث دراسة تعديل بنية المعالجات الحديثة من خلال دراسة استبدال الذاكرة المخبأة (Cache Memory) بالذاكرة Scratchpad memory (SPM)، لما للأخيرة من مواصفات تتناسب مع المتطلبات الزمنية لأنظمة CPS هذا من ناحية، بالإضافة إلى فتح المجال لتطوير خوارزميات الجدولة والاستبدال بحسب التطبيق من ناحية أخرى. ستتم المقارنة بين الذاكرتين عن طريق تصميم ومحاكاة نموذجين لمعالجين يتضمن الأول ذاكرة مخبأة والثاني ذاكرة SPM باستخدام برنامج Quartus وبرنامج ModelSim.

الكلمات المفتاحية: CPS ، الأنظمة المضمنة، الذاكرة المخبأة، SPM،Quartus ، ModelSim .

*استاذ مساعد في قسم هندسة النظم الحاسوبية والالكترونية_ كلية هندسة تكنولوجيا المعلومات والاتصالات_ جامعة طرطوس_ سورية.
**مدرس في قسم هندسة النظم الحاسوبية والالكترونية_ كلية هندسة تكنولوجيا المعلومات والاتصالات_ جامعة طرطوس_ سورية. ***طالبة ماجستير في قسم هندسة النظم الحاسوبية والالكترونية_ كلية هندسة تكنولوجيا المعلومات والاتصالات_ جامعة طرطوس_ سورية.

مجلة جامعة طرطوس للبحوث والدراسات العلمية_ سلسلة العلوم الهندسية المجلد (3) العدد (٥) 2019

Tartous University Journal for Research and Scientific Studies- Engineering Sciences Series Vol. (3) No. (5) 2019

Using of general purpose processor in cyber-physical systems throught SPM memory

Razan Akabati *
Susi Saleh **
Mohammed Molhem ***

(Received 16 / 6 / 2019 . Accepted 12 / 9 / 2019)

ABSTRUCT

The cyber-physical systems (CPS) refers to a new generation of embedded systems, which integrate the computational, network and physical processes. The embedded computing and networks manage the physical processes, and the feedback loops link the physical processes to computing system. Therefore, the implementation of CPS's system requires a significant harmonization and synchronization between these three different types of systems.

The use of general-purpose processors with their well-known architecture in the design of CPS systems will not be feasible, due to its inability to capture the time specifications of CPS. This research includes the study of the modification of the structure of processors through the study of replacement cache memory with scratchpad memory SPM, because the SPM's specification conforms to the time requirements of CPS. In addition, the ability to development a new scheduling and replacement algorithms as application requirements.

The two memories will be compared by designing and simulating two modules for processors, that the first processor includes cache memory and the second includes SPM using QUARTUS and ModelSim.

Key Words: CPS, embedded systems, cache memory, scratchpad memory, Quartus, ModelSim

* Associated Professor, Faculty of Information technology and Communications Engineering Tartous University- Syria.

** Assistant Professor, Faculty of Information technology and Communications Engineering/Tartous University- Syria.

** Postgraduate Student, Faculty of Information technology and Communications Engineering - Tartous University-Syria.

المقدمة:

طُرحت فكرة أنظمة CPS أول مرة عام 2006 من قبل Helen Gill [1] في المؤسسة الوطنية للعلوم في الولايات المتحدة (National Science Foundation). حيث تمّ التأكيد أن الإمكانات الاقتصادية والاجتماعية على المستوى الأكاديمي والصناعي لهذه النظم أكبر بكثير مما تم تحقيقه حتى الآن، وتُبذل جهود كبيرة في جميع أنحاء العالم لتطوير هذه التكنولوجيا.

تُشكل القضايا الزمنية في العمليات الفيزيائية الركيزة الأساسية لهذا النوع من العمليات، وبما أن الزمن في هذه العمليات مستمر وحقيقي (real time) يتم توصيف هذه الأنظمة باستخدام المعادلات التفاضلية. أما في العمليات الحاسوبية فالزمن متقطع، وغير حقيقي (model time)، وأساسه نبضة الساعة للنظام الحاسوبي، ويتم التركيز في هذه العمليات على الأداء الصحيح من دون مراعاة القضايا الزمنية. بالإضافة إلى أن تقنيات الشبكات ذات الأداء الأفضل في يومنا هذا لا يمكن الاعتماد عليها بأداء موثوق في الزمن الحقيقي، نتيجة الزمن الذي تحتاجه هذه الشبكات لنقل البيانات عبرها. ونتيجة لهذه الاختلافات في طبيعة الأنظمة التي تشكل أنظمة CPS كان من الطبيعي ظهور العديد من المشاكل في أثناء تنفيذ النظم الفيزيائية المحوسبة.

لذلك تتطلب نظم CPS الحديثة إجراء تعديلات في النظم الحاسوبية وتزامن أكثر إحكاماً مما هو موجود في الأنظمة الأخرى، والزمن هو العنصر المشترك بين الحوسبة والنظم الفيزيائية في CPS، والترابط الصحيح ضروري للحصول على الوظيفة المطلوبة في CPS [2].

ومن ناحية أخرى، نتيجة تطور تكنولوجيات الحوسبة والإلكترونيات نشأت لدينا مشكلة تعرف باسم مشكلة عنق الزجاجة (Bottleneck)، والمقصود بها اختلاف السرعة بين تنفيذ العمليات في المعالج وتخزين البيانات في الذواكر. وبسبب الفجوة المتزايدة بين سرعة المعالج وأداء الذاكرة، ازدادت مشكلة عنق زجاجة بشكل كبير في الحوسبة، فأصبح الوصول إلى الذاكرة لسحب للقراءة أو الكتابة أبطأ بكثير من سرعة تنفيذ العمليات في المعالج، الأمر الذي يفرض على المعالج الانتظار حتى يكتمل الوصول إلى الذاكرة وجلب البيانات أو تخزينها. فكان لابد من إيجاد حل لهذه المشكلة مما أدى إلى ظهور الذاكرة المخبأة (cache memory) لسد تلك الفجوة وتسريع الوصول إلى الذاكرة. وأصبح من المستحيل الاستغناء عن وجود مثل هذا النوع من الذواكر في المعالجات الحديثة.

من ناحية أخرى، نتيجة الخواص والميزات التي تتمتع بها ذاكرة SPM تم الاهتمام بها وجعلها جذابة جداً للتطبيقات في الزمن الحقيقي، مما يجعلها البديل المثالي للذاكرة المخبأة في نظم الزمن الحقيقي بشكل عام ونظم CPS بشكل أكثر خصوصية [3].

أهمية البحث وأهدافه

لقد واجهت أنظمة CPS العديد من المشاكل منها:

- عدم وجود دلالات زمنية مستمرة ونماذج تدعم هذا النوع من التزامن في العمليات الحاسوبية.
- تقنيات الشبكات ذات الأداء الأفضل في يومنا هذا لا يمكن الاعتماد عليها بأداء موثوق في الزمن الحقيقي.

- لم يدخل الزمن كعنصر ضمن البرمجة الغرضية التوجه، لذلك جميع تكنولوجيات المكونات البرمجية المتضمنة تصاميم غرضية التوجه تم بناؤها لدعم التطابق في البرمجيات، وليس الهدف هو النظم الفيزيائية.

يتم عادة تلافي هذه المشاكل عن طريق حساب أسوأ حالة لزمن التنفيذ (Worst Case Execution Time) WCET، واستخدام أنظمة تشغيل الزمن الحقيقي RTOS (Real Time Operating System) مع سياسات جدولة قابلة للتنبؤ، ولكن ذلك يتطلب موثوقية كبيرة جداً[4]، وبالإضافة إلى ذلك أصبح حساب WCET صعباً جداً في الأنظمة المعقدة جداً.

بالإضافة إلى ذلك، تهتم أنظمة الزمن الحقيقي بالقضايا الزمنية بشكل أساسي، بينما البنى العتادية للمعالجات لا تدعم هذه القضايا.

يسهل تحقيق ضبط الزمن بالعودة إلى معالجات 8bit والتخلي عن بنى أساسية في المعالجات الحالية مثل الذواكر المخبأة والتخلي عن 40 عاماً من التقدم في لغات البرمجة وأنظمة التشغيل والشبكات. هذه الفكرة هي المشكلة في حد ذاتها؛ إذ لا يمكننا نسف الجهود المبذولة في نصف القرن الأخير[4].

سنتناول في بحثنا طريقة تعديل الذاكرة المخبأة (Cache Memory) لتتناسب مع المتطلبات الزمنية المطلوب تحقيقها في أنظمة CPS .

الدراسات المرجعية

لقد أجريت العديد من الأبحاث في مجال نظم CPS، وتحديد معوقات تطبيقها واقتراح التعديلات في النظم الحاسوبية لتطبيق نظم CPS على أرض الواقع، فكانت هذه الأبحاث من أهم الأبحاث المطروحة على مستوى العالم منذ بداية عام 2007 وحتى يومنا هذا.

قام الباحث دافيد لانغوس عام ٢٠٠٨ [3] بالمقارنة بين ذاكرة SPM والذاكرة المخبأة من ناحية المساحة والطاقة المستهلكة والقدرة على التنبؤ، كما درس بعض خوارزميات نقل محتويات ذاكرة SPM لكنه لم يتناول فكرة توظيف هذه الذواكر ضمن نظم CPS وقام بإجراء المحاكاة باستخدام ARMulator بدلاً من برنامج ModelSim و Quartus.

وفي العام ذاته قام الباحث ادوارد لي [5] بدراسة المشاكل والتحديات التي تواجه نظم CPS وصعوبة تطبيقها باستخدام بنى المعالجات المتطورة الحالية، لكنه لم يذكر في بحثه هذا كيفية معالجة التحديات المذكورة. وقد قام الباحث جوليوس روب في العام 2013 [6] بدراسة ذاكرة SPM وتحديد أهم ميزاتها وأهم التطبيقات التي من الممكن توظيف هذه الذاكرة فيها، والمقارنة بينها وبين الذاكرة المخبأة باستخدام العلاقات الرياضية من دون استخدام برامج المحاكاة المستخدمة في بحثنا هذا. وفي العام ٢٠١٥ قام الباحث ادوارد لي [7] بدراسة نشأة نظم CPS والجهود الحالية في هذا المجال والآمال المستقبلية المبنية على هذه الأنظمة، وطرح طرق لبناء نماذج للنظم بشكل عام والأنظمة المضمنة بشكل خاص، ولكن لم يقم بإجراء أي محاكاة ولم يقارن مع الذاكرة المخبأة. وقد حدد الباحثون في جامعة كاليفورنيا في العام ٢٠١٦ [2] المتطلبات الزمنية التي تحتاجها نظم CPS ولكن دون ذكر الحلول المتبعة لمعالجة ومواجهة التحديات. وفي بداية العام ٢٠١٨ قام الباحث سعود واصلي بدراسة الذاكرة SPM وطرح آلية استخدام هذه الذاكرة في المعالجات متعددة النوى ولكن الدراسة كانت تشمل دراسة خوارزميات الجدولة المطبقة في نظام التشغيل[8].

المشاكل الزمنية في نظم CPS

في نظم CPS يتم إجراء عمليات الحوسبة على البيانات القادمة من الوسط الفيزيائي ثم يتم اتخاذ الاستجابة المناسبة بناءً على القرارات التي يتخذها نظام الحوسبة للتأثير في الوسط الفيزيائي، وكل ذلك يتم عبر شبكات الاتصال بين الوسطين الحاسوبي والفيزيائي. ومن هنا نشأت أولى التحديات الرئيسة في تصميم CPS هو إنشاء مفهوم مشترك للزمن بين العالم الفيزيائي حيث يكون الزمن مستمراً والنظام الحاسوبي حيث يتم زيادة الوقت في وحدات متقطعة (بالاعتماد على نبضات الساعة الحاسوبية).

تكون أنظمة الأغراض العامة مصممة لتحسين أداء الحالة المتوسطة (average-case performance) لأنه يهمنا بشكل أساسي الأداء والحصول على النتائج الصحيحة بغض النظر عن الزمن الذي نحتاجه للاستجابة.

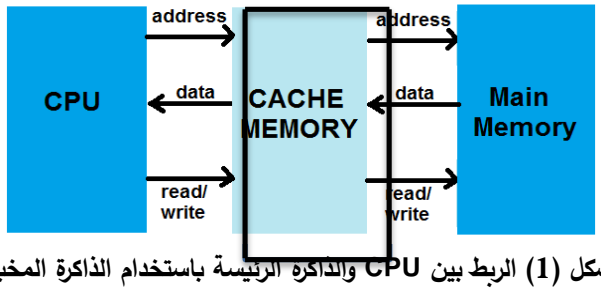
أما في نظم الزمن الحقيقي RTS (Real Time System) [9] يشكل زمن الاستجابة عنصراً مهماً جداً، لذلك نهتم بإمكانية التوقع بحساب أسوأ حالة لزمن التنفيذ WCET الذي هو ضروري لتحليل الجدولة في النظام وهو صعب للغاية في النظم ذات الأغراض العامة حيث يوجد عدة أسباب لحدوث التوقع الصحيح، ومن أهم هذه الأسباب وجود الذاكرة المخبأة وعدم القدرة على تحديد الوصول إلى الذاكرة لأن سلوكها يتعلق بحالتها الحالية. بالإضافة إلى أن تنفيذ المهام في الأنظمة المتعددة المهام يرتبط بأولوية المهمة فأصبح من الصعب جداً تحديد الحالة التي توجد فيها الذاكرة المخبأة (أي مهمة يتم تنفيذها خلال لحظة زمنية معينة). وزيادة تعقيد الأنظمة الحاسوبية أدى إلى وجود فرق كبير بين WCET الفعلي والتقدير الكبير جداً.

ونتيجة لهذه المصاعب كان لابد من التوجه في البحث إلى تطوير بنية المعالجات الحالية من الناحية العتادية لدعم المتطلبات الزمنية المطلوب تحقيقها في أنظمة CPS. سنتناول في بحثنا استبدال الذاكرة المخبأة (Cache memory) بذاكرة ScratchPad Memory(SPM[10]) لتتناسب مع متطلبات نظم CPS.

دراسة الفرق بين الذاكرة المخبأة وذاكرة SPM

١ -آلية عمل الذاكرة المخبأة

تشكل الذاكرة المخبأة [11] ذاكرة وسيطية بين وحدة المعالجة المركزية CPU والذاكرة الرئيسة، بحيث لانستطيع تبادل البيانات بين الذاكرة الرئيسة ووحدة المعالجة المركزية إلا عن طريق الذاكرة المخبأة، بحيث يتم القراءة منها والكتابة عليها ويجب أن تحتوي نسخة من البيانات الموجودة في الذاكرة الرئيسة كما هو موضح بالشكل (1).



الشكل (1) الربط بين CPU والذاكرة الرئيسة باستخدام الذاكرة المخبأة

إذا كانت المعطيات المطلوبة متاحة في الذاكرة المخبأة نسمي هذه الحالة حالة إصابة (hit)، وإلا يتم إحضار المعطيات من الذاكرة الرئيسة وتسمى حالة الفقد (miss). وفي حالة الفقد يتم أخذ نسخة من البيانات وتخزينها في موقع محدد وثابت في الذاكرة المخبأة وتسمى هذه الخوارزمية (Direct mapped)، أو تتوضع في أي مكان من

الذاكرة المخبأة وتسمى هذه الخوارزمية (Fully associative)، أو تتوضع في مكان محدد من مجموعة من المواقع وتسمى هذه الخوارزمية (set associative) .

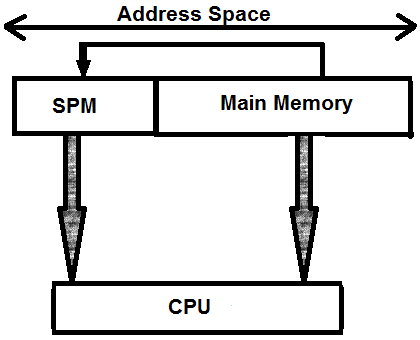
وعند امتلاء الذاكرة المخبأة ونحتاج لاستبدال البيانات ضمن أحد السطور، فإما نقوم بحذف البيانات التي تم تخزينها أولاً (FIFO)، وإما يتم حذف عشوائياً (Random)، وإما يتم حذف البيانات التي لم تُستخدم حديثاً [12] (LRU). وبما أن الذاكرة الرئيسة يجب أن تحتوي نسخة طبق الأصل عن الذاكرة المخبأة، يجب أن يتم التحديث بشكل دوري إما عند كل كتابة (Write-through)، وإما يتم تحديث بيانات الذاكرة الرئيسة عند اختيار المعطيات ليتم حذفها من الذاكرة المخبأة (Write_Back).

٢ -آلية عمل ذاكرة SPM

يُعد SPM المصطلح المختار للذاكرة التي تتم إدارتها برمجياً بحيث تقوم البرمجيات بقيادة الذاكرة وتعيد النتائج لتخزن في موقع محدد من الذاكرة، وكونها تقع على الشريحة نفسها - وعلى مقربة من- وحدة المعالجة المركزية الأساسية، فإن تأخر الوصول إليها لا يكاد يذكر بالمقارنة مع الموجود في الذاكرة الرئيسة. ومن الجدير ذكره أن فضاء العناوين لذاكرة SPM مختلف عن فضاء عناوين الذاكرة الرئيسة؛ وبالتالي يمكن لوحدة المعالجة المركزية أن تتابع حساباتها بينما يتم نقل البيانات من الذاكرة الرئيسة إلى SPM أو العكس بالعكس كما هو موضح في الشكل (٢)، ويتم بناء هذه الذاكرة من خلايا SRAM.

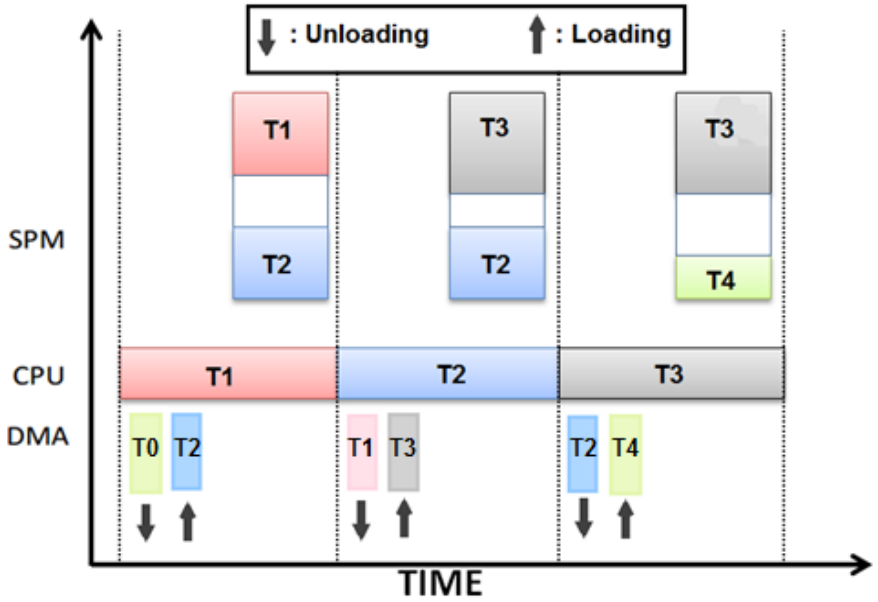
بشكل عام يجري تقسيم SPM إلى ثلاثة أقسام متغيرة الحجم:

- قسم خاص لنظام التشغيل بالإضافة إلى بعض المكتبات التي قد تحتاجها المهام الحرجة.
- قسمان لتنفيذ المهام ويقوم نظام التشغيل بإدارة هذا القسم .



الشكل (٢) فضاء العناوين في SPM

يوضح الشكل (٣) الآلية المستخدمة لتحميل المهام ضمن ذاكرة SPM [13].



الشكل (٣) آلية تحميل المهام إلى SPM

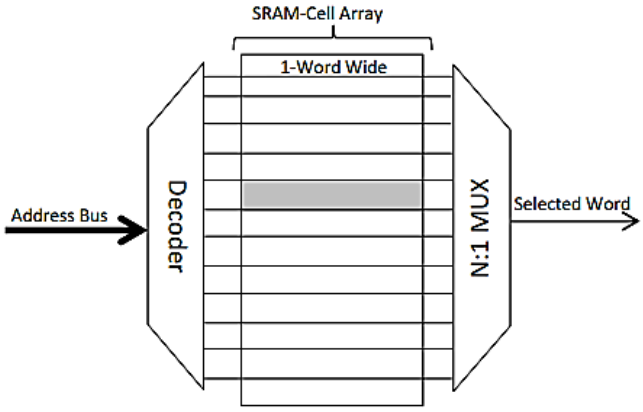
كما هو موضح في الشكل (٣) في أثناء تنفيذ المهام الحرجة تقوم DMA(Direct memory access) بتحميل رمز وبيانات المهمة الحرجة التالية إلى SPM. ويقوم النظام بإلغاء المهمة الحرجة المنفذة سابقاً وإعادة كتابة البيانات في الذاكرة الرئيسة. على سبيل المثال T2 يتم تحميلها إلى SPM في أثناء تنفيذ T1 لذلك يتم تنفيذها مباشرة بعد الانتهاء من تنفيذ T1 بدون أي تأخير. ويتم حذف T1 من ذاكرة SPM بعد الانتهاء من تنفيذها وبالتالي نستطيع تحميل مهمة جديدة وهكذا.

3 –المقارنة بين الذاكرة المخبأة وذاكرة SPM

يتم بناء الذاكرة عادة على رقاقة (SPM أو الذاكرة المخبأة) باستخدام خلايا ذاكرة الوصول العشوائي الثابتة SRAM [١٤]، هي نوع من الذواكر التي تستخدم دارة الإغلاق الثنائي (flip–flop) لتخزين كل بت.

يظهر الشكل (٤) بنية ذاكرة SPM [13] بحيث نلاحظ أنها عبارة عن دارة رقمية بسيطة، وبالتالي تحتل مساحة أصغر وتستهلك طاقة أقل ولا يتطلب الوصول إلى كلمة الذاكرة سوى مفكك تشفير Decoder وناخب Multiplexer.

يسمى الوصول إلى الذاكرة واختيار كلمة معينة بالفهرسة indexing. في أثناء الوصول، يتم تحديد خلايا SRAM المستهدفة ثم قراءتها أو كتابتها.



الشكل (٤) بنية ذاكرة SPM

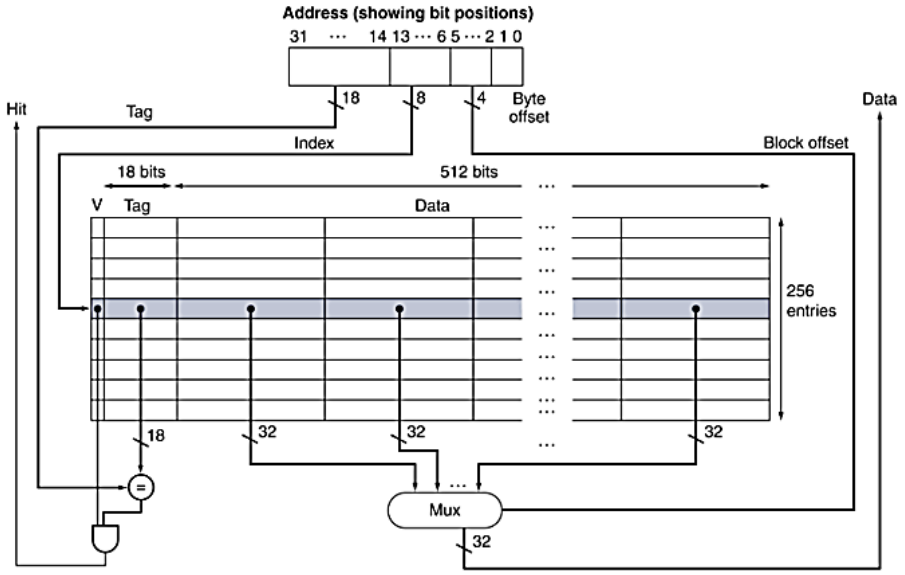
من ناحية أخرى، فإن بنية الذاكرة المخبأة أكثر تعقيدًا كما هو مبين في الشكل(٥). وهي تتألف من ذواكر متعددة للبيانات، وكلها مفهرسة في الوقت نفسه [١٥]. بالإضافة إلى ذلك، تتم فهرسة مصفوفات العلامة tag array وبت الصلاحية valid bit لكل وصول، مما يستهلك طاقة أكبر. علاوة على ذلك، يتم تنفيذ اختيار الكلمة المعنونة في خطوتين:

أولاً : يتم تحديد خط (line) في الذاكرة المخبأة بالكامل وذلك بالطريقة نفسها المستخدمة في SPM عن طريق (N: 1 MUX).

ثانياً : تحديد إزاحة الكتلة (block) (جزء من العنوان) أي تحديد كلمة واحدة من سطر الذاكرة المخبأة. هذه الآلية في الوصول إلى الكلمة المعنونة يجعل الذاكرة المخبأة أبطأ من ذاكرة scratchpad. وتستخدم الذاكرة المخبأة مواقع ذاكرية أكبر لتخزين العلامات(tags) والبتات الصالحة (valid bits).

يؤدي الوصول السريع، والمساحة الأصغر، والطاقة المنخفضة [14]، ووقت الوصول الذي يمكن التنبؤ به إلى جعل ذواكر scratchpad جذابة للأنظمة المضمنة وبخاصة في الزمن الحقيقي. حيث عند استخدام الذاكرة المخبأة في معالج الأغراض العامة، يحتوي نظام التشغيل جميعَ التعليمات المسؤولة عن النظام ويتم جلب المعطيات من الذاكرة الرئيسة (إذا لم تكن موجودة في الذاكرة المخبأة) بعد وضع نسخة منها في الذاكرة المخبأة. بينما عند استخدام ذاكرة SPM يتم تحميل المهام عند زمن التشغيل إلى الذاكرة الرئيسة ثم تحميل المهام بالتتابع إلى SPM.

يتم التحكم في الذاكرة المخبأة بواسطة العتاد الصلب (hardware)، فلا يجب إجراء أي تعديلات على العتاد البرمجي (software) مما يؤدي إلى مشاكل القدرة على التنبؤ في أنظمة الزمن الحقيقي، بينما يتم التحكم و تخصيص بيانات في scratchpad بواسطة البرنامج (software)، وهذا يؤدي إلى إمكانية عالية للتنبؤ تجعل من الممكن استخدامها في أنظمة الزمن الحقيقي.



الشكل (٥) بنية الذاكرة المخبأة

ومع ذلك، فإن إدارة SPM أكثر صعوبة في إدارة الذاكرة المخبأة؛ لأنه يتم إدارتها برمجياً (software) وبالتالي يحتاج المبرمجون إلى معرفة منصة الأجهزة التي يعملون عليها. في أنظمة تعدد المهام، يصبح الأمر صعباً للغاية لأن المبرمجين بحاجة إلى إدارة SPM، ومزامنة أعمالهم وفقاً لذلك. من ناحية أخرى، تتم إدارة الذواكر المخبأة ذاتياً على مستوى العتاد الصلب (hardware) ولا تتأثر بنوعية العتاد البرمجي (software). بمعنى آخر، تكون الذاكرة المخبأة شفافة للمبرمجين. وبالتالي، يمكن نقل التطبيقات إلى أنظمة أساسية جديدة بسهولة أكبر. يظهر الجدول (1) أوجه الاختلاف بين ذاكرة SPM والذاكرة المخبأة.

الجدول (١) أوجه الاختلاف بين الذاكرة المخبأة وذاكرة SPM

	ذاكرة SPM	الذاكرة المخبأة
البنية والحجم	صغيرة لأن بنيتها بسيطة	أكبر لأن بنيتها معقدة
التخزين	تخزن جزء من البيانات الحرجة	تخزن نسخة من البيانات الموجودة في الذاكرة الرئيسة
فضاء العناوين	مختلف عن فضاء العناوين في الذاكرة الرئيسة	نفس فضاء العناوين في الذاكرة الرئيسة
البحث عن البيانات ضمنها	إصابة دوماً (Only hit)	إصابة أو فقد (hit or miss)
الادارة	تتم إدارتها باستخدام العتاد البرمجي	تتم إدارتها باستخدام العتاد الصلب
زمن الولوج	يمكن توقعه	غير قابل للتوقع

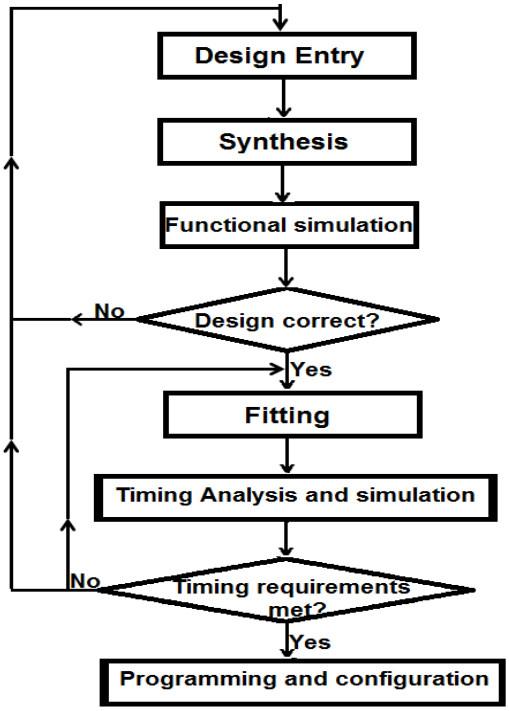
طرائق البحث وموارده

١ -توصيف برنامج Quartus

يعد برنامج Quartus من أشهر برامج شركة Altera [16]، وهو عبارة عن بيئة متكاملة لتصميم النظام على رقاقة قابلة للبرمجة (SOPC System-on-a-programmable-chip) مثل FPGA,CPLD [17].

ويتميز بطرق إدخال بسيطة للتصميم باستخدام إحدى لغات وصف العتاد الصلب مثل VHDL أو Verilog، ومعالجة سريعة وبساطة في التصميم، وسهولة إدخال الأوامر إما باستخدام الأوامر المكتوبةscripting أو باستخدام الواجهات GUI، وإمكانية إجراء أي تعديل بسهولة على التصميم، وأدوات للمحاكاة والتحليل الزمني والمنطقي، وإمكانية استخدام الملفات الناتجة عن أدوات تحليل ومحاكاة في البرنامج.

والتصميم باستخدام برنامج Quartus [17] كما هو موضح بالشكل (6) يبدأ بإدخال التصميم (Design Entry) وتتضمن هذه المرحلة توصيفاً للنظام المرغوب إما باستخدام لغة وصف الكيان الصلب مثل , VHDL Verilog وإما باستخدام ما يسمى المخطط الصندوقي أو (Block Diagram).ثم تأتي مرحلة التأليف (Synthesis) وهي أداة مساعدة باستخدام الحاسب تعطي بناء على توصيف النظام مخططاً له يسمى netlist وهو يمثل العناصر المنطقية التي يتكون منها التصميم والوصلات بين هذه العناصر. ثم مرحلة المحاكاة الوظيفة (Functional Simulation) و يتم فيها فحص الدارة للتحقق من وظيفتها (بغض النظر عن الزمن). ثم مرحلة الترجمة (Fitting) بحيث يطابق العناصر المنطقية في الnetlist مع عناصر منطقية على الرقاقة الحقيقية كما وتحدد أسلاك التوجيه على الرقاقة لتحقيق الوصلات المناسبة بين العناصر المختارة. ثم مرحلة التحليل الزمني (Timing Analysis) ويتم فيه تحديد تأخير الانتشار للإشارة خلال المسارات المختلفة في الدارة المحددة باستخدام الـ Fitter وتحلل لمعرفة الأداء المتوقع من هذه الدارة. ثم تتم عملية المحاكاة الزمنية(Timing Simulation) ويتم فيها التحقق من أداء الدارة زمنياً ووظيفياً. أما المرحلة الأخيرة فهي البرمجة والتهيئة (Programing and configuration) ويتم فيها تطبيق الدارة المصممة والمختبرة على الرقاقة الفيزيائية وذلك عن طريق البرمجة للقواطع بحيث تحقق الوصلات اللازمة بين العناصر الالكترونية.

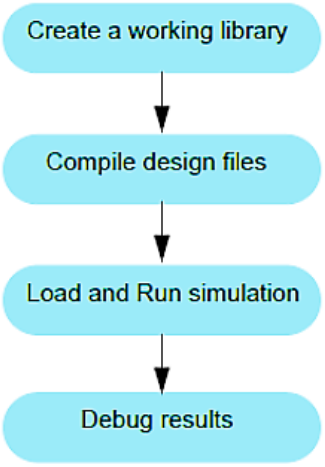


الشكل (٦) أساسيات تدفق التصميم في برنامج QUARTUS

٢ –توصيف برنامج ModelSim

برنامج ModelSim [18] هو أداة محاكاة للتصاميم الموصوفة بلغات VHDL و Verilog و Verilogsystem كما يمكن أن يكون التصميم موصوفا بعدّة لغات أو ما يسمى بالتصميم بلغات مختلطة .

يتم تصميم البرنامج كما هو موضح بالشكل (٧) بحيث يبدأ التصميم بتوليد مكتبة العمل (Creating the Working Library) بحيث توضع جميع التصاميم في برنامج modelsim داخل مكتبة افتراضية تسمّى "work" ، يضع المجّمع(compiler) بشكل افتراضي جميع وحدات التصميم داخل هذه المكتبة . ثم مرحلة تجميع التصميم (Compiling Your Design) بحيث أنه بعد توليد مكتبة العمل ، يتم تجميع التصميم ضمنها ، وتتوافق صيغة المكتبة مع جميع المنصات (platform)، بحيث يمكن للمصمّم أن يطبّق التصميم على أي منصّة دون الحاجة الى إعادة التجميع .ثم يحمّل المحاكي (simulator) بالتصميم وتشغّل عملية المحاكاة ، حيث يتم تحديد المستوى الأعلى(top–level module) ل (Verilog)، أو الكيان /البنية entity/architecture pair بالنسبة إلى لغة (VHDL). وبعد تحميل التصميم بنجاح يوضع زمن المحاكاة على الصفر، ونضغط على أمر التشغيل لبدأ عملية المحاكاة. ثم مرحلة تنقيح النتائج(Debugging Your Results) فإذا لم نحصل على النتائج المتوقعة بعد عملية التشغيل، يمكننا استخدام منقّح modelsim لحل المشكلة.



الشكل (٧) أساسيات تدفق التصميم في برنامج ModelSim

التطبيق العملي

١ –تصميم معالج يحتوي ذاكرة SPM

من خلال الدراسة السابقة، تم تصميم معالج يحاكي نموذج لمعالج يحوي ذاكرة SPM،بحيث يحوي وحدة حساب ومنطق (ALU) يمكنها تنفيذ العمليات الحسابية والمنطقية (Add, Subtract, Or, Xor, And, Not, Read, Write, Load, Compare, Shift Left, Shift Right, Jump/Branch, Jump/Branch conditionally).

تم بناء معالج ليقوم بمجموعة من العمليات المتسلسلة وهي (ADD, AND, NOT, SHL)، وبماأن المعالج الذي يحتوي ذاكرة SPM يخزن نظام التشغيل على الذاكرة سوف نخزن هذه التعليمات في رتل على شكل عنوان من ١٩ بت مقسم كما هو موضح في الجدول (٢).

الجدول (٢) تقسيم التعليمة المكونة من ١٩ بت

	١٨	١٥ ١٤	١٠ ٩ ٧	٥ ٤ ٠
RRR	**Opcode**	**rD**	**Ra**	**Rb**
RRs	**Opcode**	**unused**	**Ra**	**Rb**
RRd	**Opcode**	**rD**	**rA**	**Unused**
R	**Opcode**	**rD**	**Unused**	
RImm	**Opcode**	**rD**	**unused**	**8-bit Immediate value**
Imm	**Opcode**	**unused**	**Unused**	**8-bit Immediate value**

بحيث إن opcode هو ترميز التعليمة الذي يختلف من تعليمة لأخرى. rD هو الموقع من الذاكرة الذي ستخزن فيه النتيجة. rA وrB هي المواقع المصدر التي سنسحب منها البيانات ، والتي سيتم إجراء العمليات في وحدة المعالجة المركزية على هذه البيانات.

بما أن فضاء العنوان مشترك بين الذاكرة الرئيسة وذاكرة SPM، سنستخدم ٥ بت لترميز كل موقع ذاكري، بحيث إن العناوين بين 00000 و ٠٠١١١ هي العناوين المحجوزة للذاكرة SPM أي ٨ مواقع، بينما العناوين بين 01000 و ١١١١١ هي العناوين المخصصة للذاكرة الرئيسة، ويتم تحديد موقع البيانات المطلوبة باستخدام مقارن .

يشير العمود الأول من الجدول (3–1) إلى نوع التعليمة، بحيث يشير الرمز RRR إلى بعض التعليمات التي تحتاج إلى ثلاثة مواقع ذاكرية: موقع مصدر لجلب البيانات وموقع هدف لتخزين النتيجة مثل تعليمة الجمع والطرح، وبعض التعليمات الأخرى تحتاج إلى موقع مصدر وموقع هدف RRd مثل تعليمة Not، وبعض التعليمات تحتاج إلى موقعي مصدر نرمز لها RRs مثل عملية الكتابة في الذاكرة، وتعليمات تحتاج إلى موقع مصدر فقط R مثل القفز إلى موقع محدد، وتعليمات تحتاج إلى موقع هدف وقيمة فورية RImm مثل تعليمة load، وتعليمات تحتاج إلى قيمة فورية فقط Imm مثل تعليمة القفز إلى قيمة فورية. نستخدم عبارة **unused** للدلالة على البتات من العنوان غير المستخدمة (غير محجوزة). أما **8–bit Immediate value** فهي للدلالة على قيمة فورية من ٨ بت.

وبناءً عليه تكون قيم التعليمات المخزنة في الرتل موضحة بالجدول (٣).

الجدول (٣) التعليمات المخزنة في الرتل

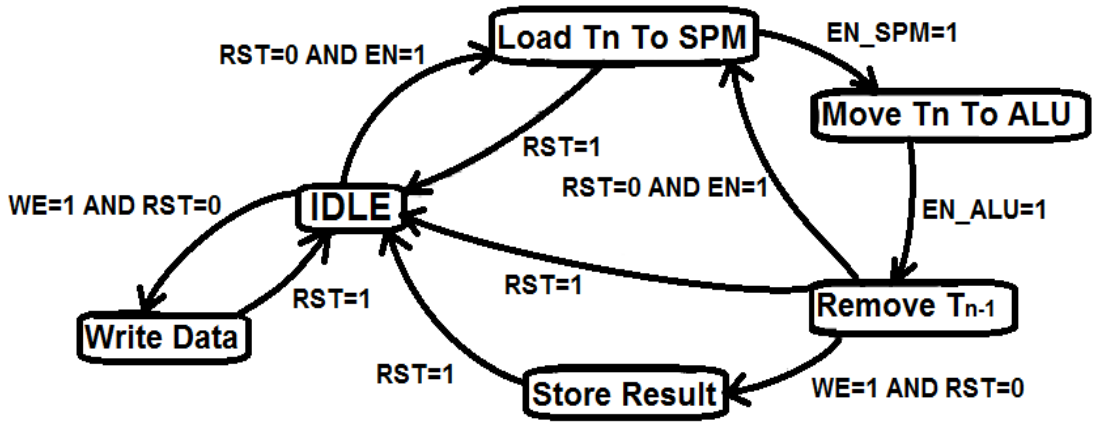
التعليمة بالصيغة الست عشرية	Opcode	موقع المصدر الاول	موقع المصدر الثاني	مسجل الهدف
٥١٤٤٣	١٠١٠ تعليمة إزاحة يسارية SHL	٠٠٠١٠	٠٠٠١١	٠٠١٠١
٢٩٨٠١	٠١٠١ تعليمة Not لمحتويات المسجل الأول	٠٠٠٠٠	–	٠٠١١٠
21C43	٠١٠٠ تعليمة جمع منطقي	٠٠٠١٠	٠٠٠١١	٠٠١١١
٠١٠٠١	٠٠٠٠ تعليمة الجمع الحسابي	٠٠٠٠٠	٠٠٠٠١	٠٠١٠٠

نقوم بتصميم الذاكرة الرئيسة وهي عبارة عن ذاكرة من ٢٤ موقعاً يتسع كل منها ١٦بت تخزن بها مجموعة من القيم المطلوب استخدامها في العمليات الحسابية والمنطقية، تقوم بإرسال البيانات المطلوبة إلى ذاكرة SPM لاستخدامها مباشرة من قبل وحدة الحساب والمنطق، وذلك بناء على إشارة التحكم القادمة من رتل التعليمات.

نقوم بتصميم ذاكرة SPM من 8 مواقع بحيث تقوم باستقبال البيانات من الذاكرة الرئيسة لتكون موجودة حتماً في ذاكرة SPM عند طلبها من وحدة الحساب والمنطق.

تقوم وحدة الحساب والمنطق بحساب النتيجة وتخزينها في الموقع الهدف المحدد في الجدول (٣).

كما أننا سننتقل بين الحالات الممكنة لذاكرة SPM كما هو موضح في الشكل (٨).



الشكل (٨) مخطط الحالة لعمل SPM

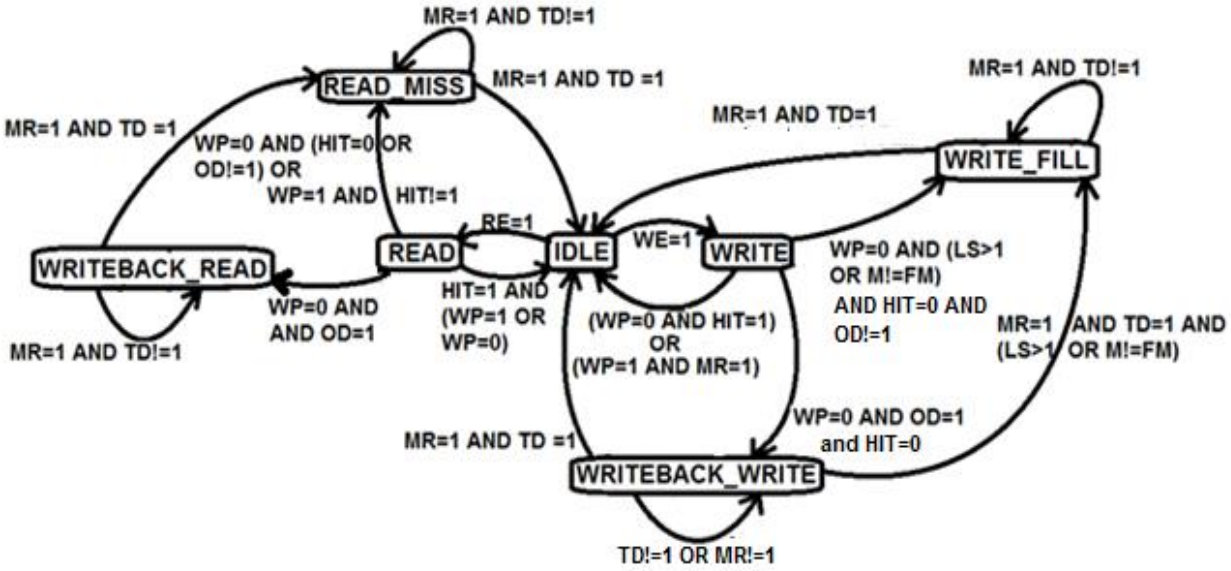
كما هو مبين في الشكل (٨) تكون الذاكرة في بداية العمل متهيئة بالقيم الصفرية في جميع مواقعها أي في حالة IDLE، وعند التفعيل EN=1، وهي إشارة تحكم قادمة من Main Memory؛ تنتقل جميع البيانات المتعلقة بالعملية Tn إلى SPM حيث إن (n عدد طبيعي يأخذ قيم ابتداء من ١) ،وعند تفعيل EN_SPM=1 تنتقل البيانات المتعلقة بالعملية Tn من SPM إلى ALU، وفي أثناء تنفيذ العملية Tn في وحدة ALU يتم حذف جميع البيانات المتعلقة بالتعليمة Tn−1 من SPM، هذا يتيح مكاناً جديداً لتخزين البيانات المتعلقة بالعملية الجديدة، يتم تخزين نتائج العمليات في SPM بعد تفعيل خط الكتابة WE=1 .كما يمكننا الكتابة في SPM بشكل منفصل عن الذاكرة الرئيسة بتفعيل خط الكتابة WE=1. يتم تهيئة الذاكرة دوماً باستخدام خط RST=1 فتعود SPM إلى الحالة الأساسية أي جميع المواقع متهيئة بالقيم الصفرية.

٢ –تصميم معالج يحتوي ذاكرة مخبأة

سيتم تصميم معالج بصفات المعالج السابق نفسها، بحيث إن الذاكرة الرئيسة مكونة من ٣٢ موقعاً كل منها يتسع ١٦بت، والتعليمة لها الصيغة السابقة نفسها، ولكن سنستخدم الذاكرة المخبأة بدل ذاكرة SPM.

تتميز الذاكرة المخبأة التي صممناها بأنه من الممكن تحديد عرض باص المعطيات وعرض باص العنوان، وكذلك يمكننا تحديد سياسة التوضع (Mapping) عن طريق الثابت ASSOC_BITS، ويمكننا تحديد خوارزمية الاستبدال عن طريق تغيير الثابت REPLACEMENT (٠: LRU، ١:MRU، ٢:FIFO ، ٣:PLRU)، ويمكننا تحديد طريقة الكتابة باستخدام الثابت WRITE_POLICY (٠: write–back ، ١: write–through) .

كما أننا سوف ننتقل بين الحالات الممكنة للذاكرة المخبأة باستخدام مخطط الحالة المبين في الشكل (٩). بحيث إن HIT هي وجود البيانات المطلوبة ضمن الذاكرة المخبأة، WP وهي سياسة الكتابة(تعديل البيانات ضمن الذاكرة الرئيسة والذاكرة المخبأة) بحيث WP=1هي Write_through و WP=0 هي Write_Back ، DB(Dirty Bit) هي بتات إضافية يتم إضافتها بحيث إذا كانت DB=1 للدلالة على أنه قد تم تعديل في الذاكرة وإذا DB=0 أي لم يتم التعديل، و OD(Oldest Dirty) هي بتات إضافية يتم إضافتها بحيث إذا كانت OD=1 ننتقل إلى Write_back ، أما MR(MREADY) فهي تدل على جاهزية الذاكرة الرئيسة إذا كانت MR=1، والإشارة TD(Transfer_Done) أي عملية نقل البيانات بين الذاكرتين قد انتهت إذا كانت TD=1، بينما LS(LINE_SIZE) تدل على عدد الأسطر المراد كتابتها فإذا كانت LS>1 أي نبدأ بالكتابة ونقل البيانات بين الذاكرتين .

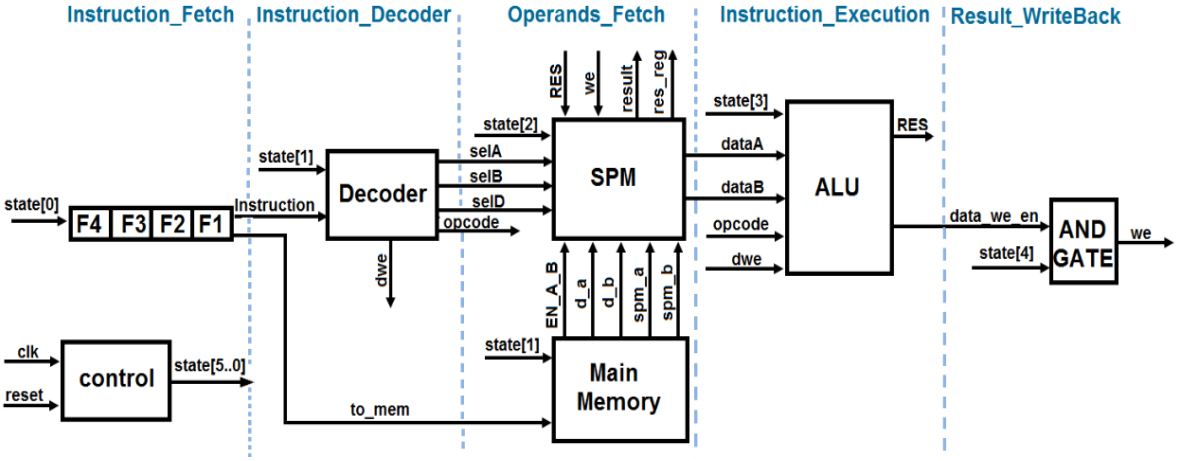


الشكل (٩) مخطط الحالة للذاكرة المخبأة

يتم تصميم الذاكرة الرئيسة مكونة من ٣٢ موقعاً يتسع كل منها ١٦ بت، لا تتعامل الذاكرة الرئيسة بشكل مباشر مع بقية الوحدات بل يتم نقل البيانات منها أو إليها عن طريق الذاكرة المخبأة حكماً. وبذلك نحتاج إلى دورات ساعة إضافية كما سنرى في أثناء المحاكاة.

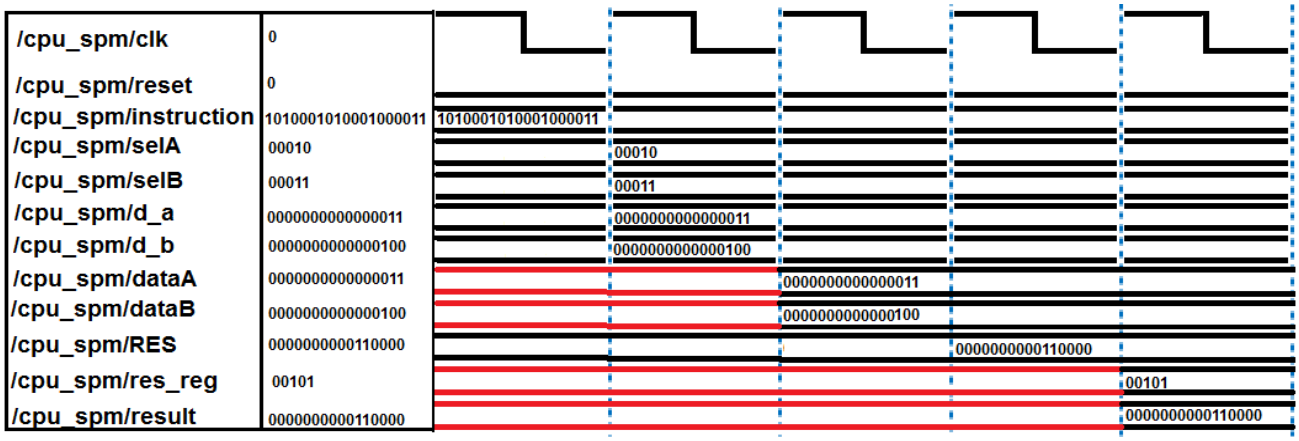
٣ –المحاكاة والنتائج

نقوم بتصميم المعالج الذي يحتوي ذاكرة SPM الموضح مخططه الصندوقي في الشكل (١٠) والذي يبين مراحل تنفيذ أية عملية.

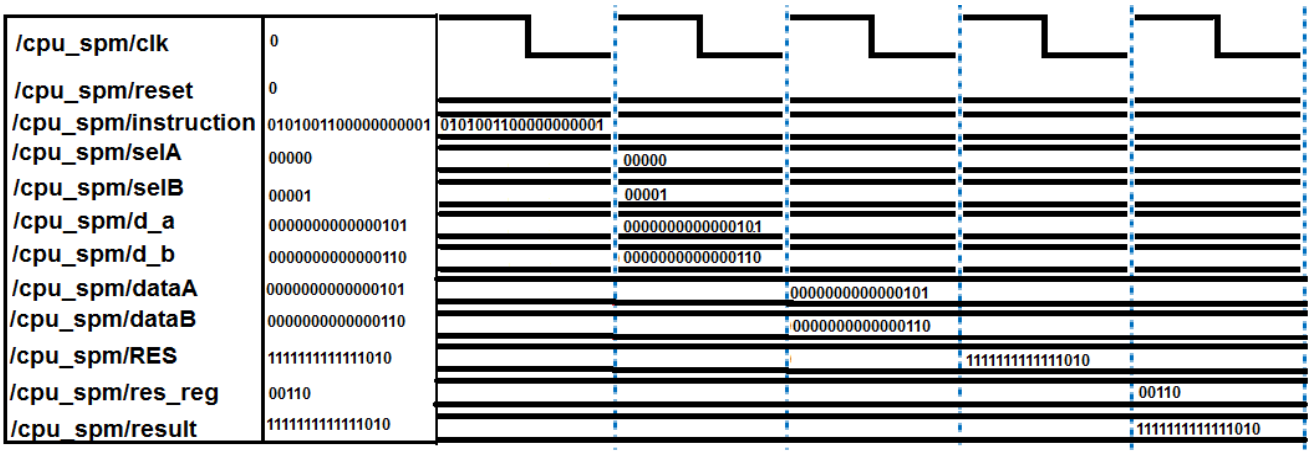


الشكل (١٠) المخطط الصندوقي للمعالج الذي يحتوي SPM

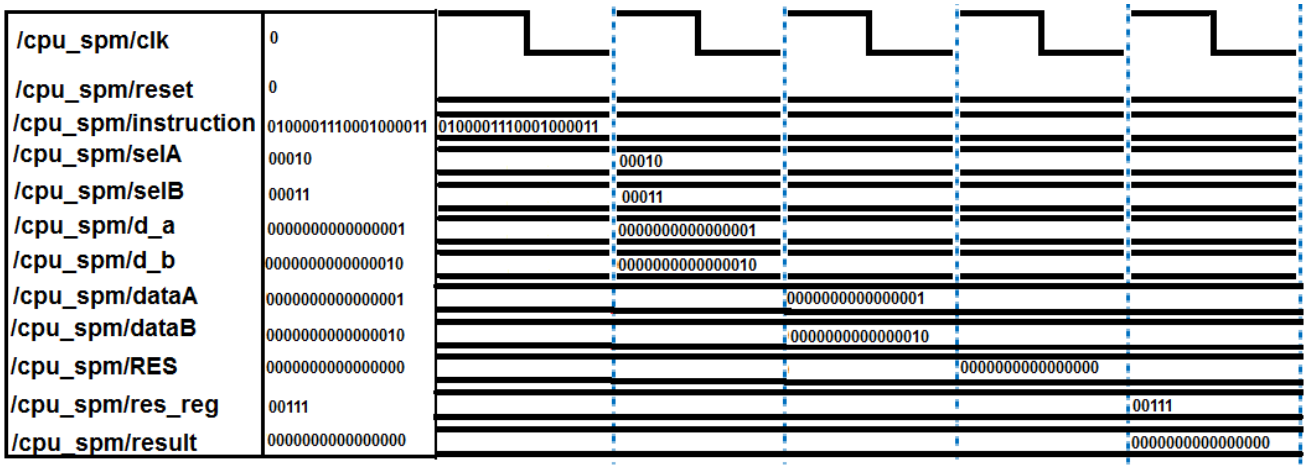
وبعد إجراء المحاكاة باستخدام برنامج ModelSim تظهر النتائج، بحيث يوضح الشكل (١١) عملية الإزاحة المطبقة على المعالج، والشكل (١٢) عملية النفي المنطقي NOT ، والشكل (١٣) عملية الجمع المنطقي AND، والشكل (١٤) عملية الجمع الحسابي ADD.



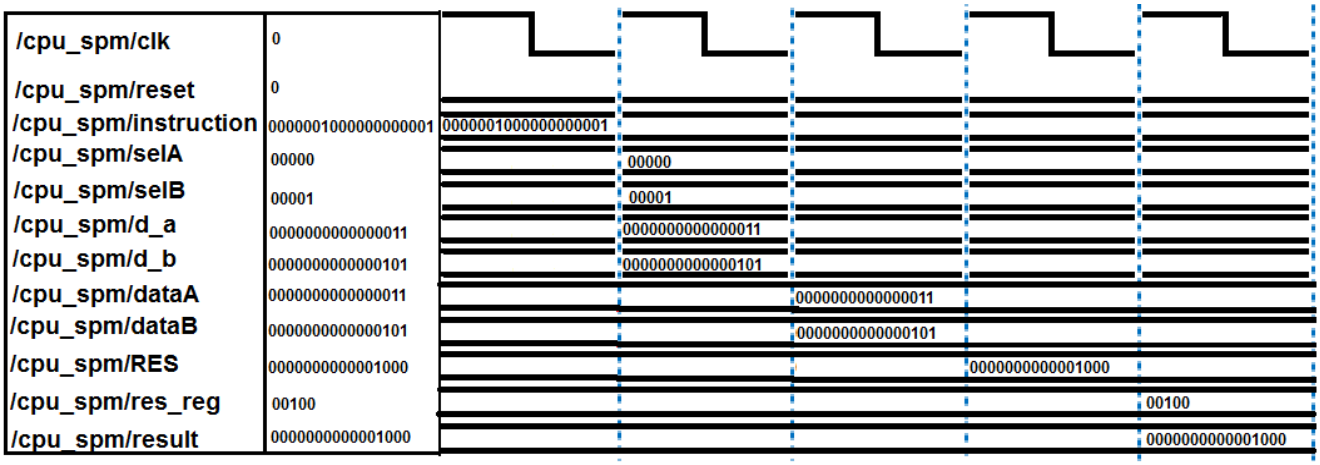
الشكل (١١) إزاحة A بمقدار B



الشكل (١٢) عملية العكس المنطقي NOT لقيمة A



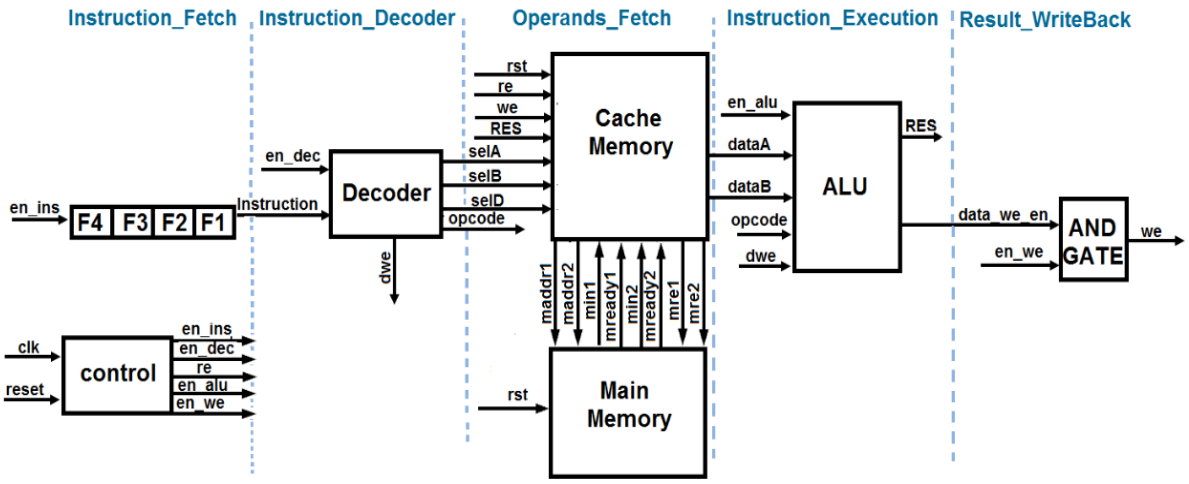
الشكل (١٣) عملية الجمع المنطقي AND للقيمتين A و B



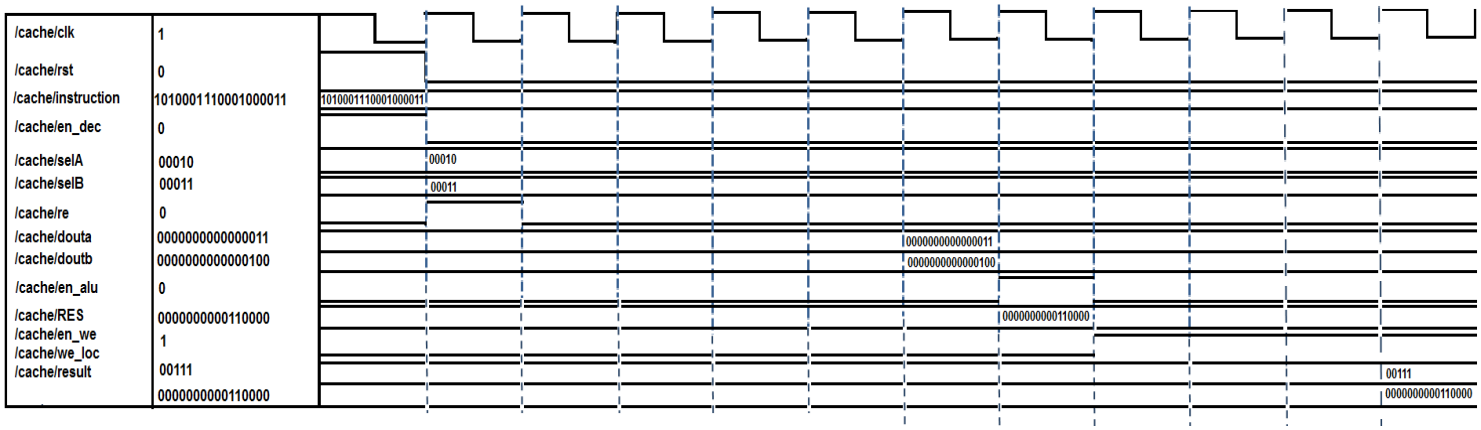
الشكل (١٤) عملية الجمع الحسابي ADD للقيمتين A وB

نلاحظ من المخططات السابقة أن أية عملية مهما كانت من العمليات السابقة، تحتاج إلى خمس نبضات ساعة للحصول على النتيجة وإعادة تخزينها، هذا يعني أننا نستطيع التنبؤ بالمدة التي نحتاجها لتنفيذ أية عملية باستخدام معالج يضم ذاكرة SPM، وهذا من أهم الشروط الواجب توافرها في معالج لنستطيع استخدامه في نظم CPS.

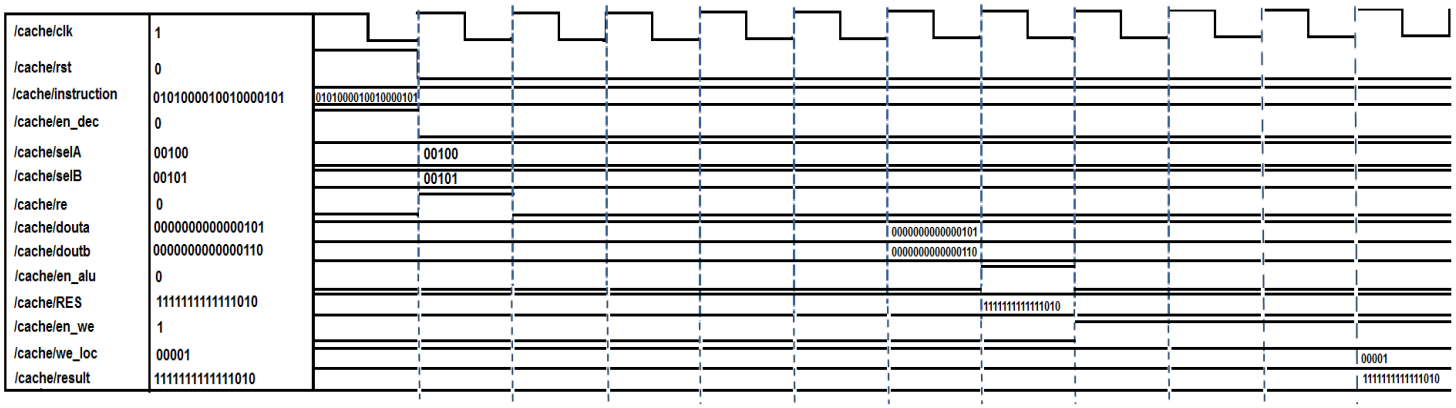
وقد جرى تصميم المعالج الذي يحوي الذاكرة المخبأة، وكما هو موضح مخططه الصندوقي في الشكل (١٥)،وتم إجراء عملية المحاكاة له ، بحيث يوضح الشكل (١٦) عملية الإزاحة المطبقة على المعالج، والشكل (١٧) عملية النفي المنطقي NOT ، والشكل (١٨) عملية الجمع المنطقي AND، والشكل (١٩) عملية الجمع الحسابي ADD.



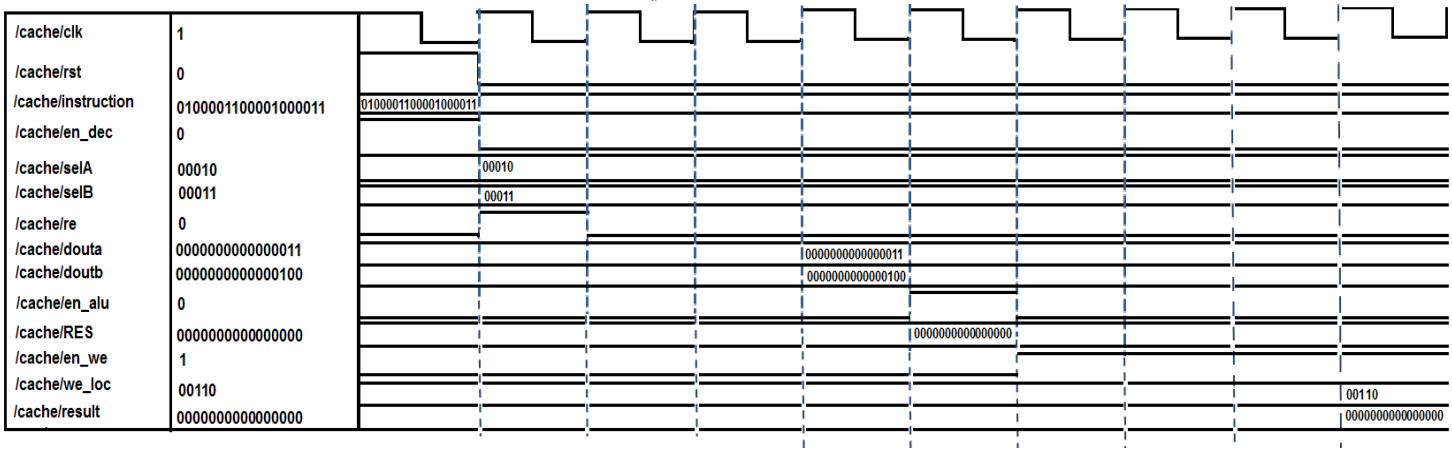
الشكل(١٥) المخطط الصندوقي للمعالج الذي يحتوي ذاكرة مخبأة



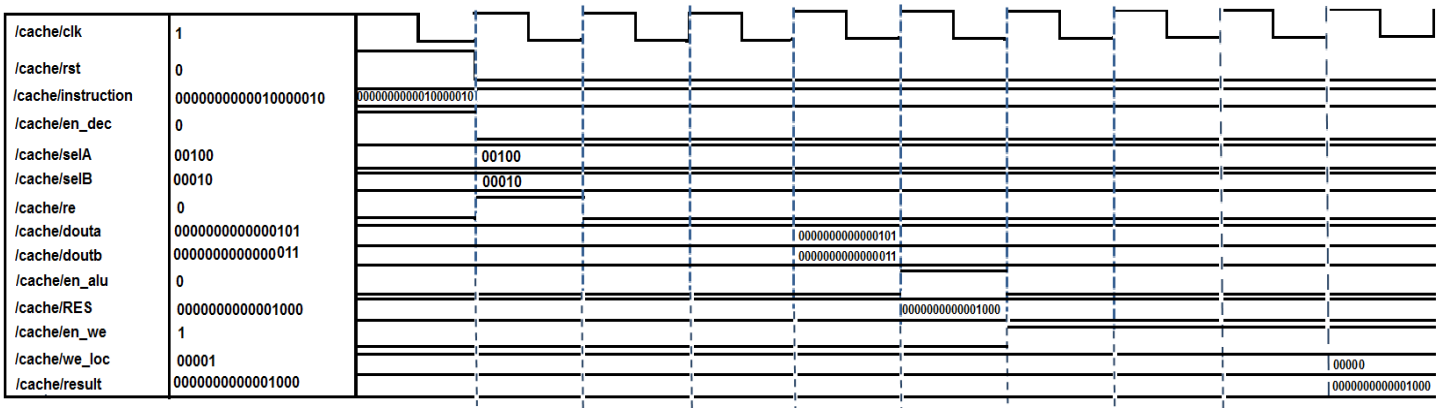
الشكل (١٦) إزاحة A بمقدار B



الشكل (١٧) عملية العكس المنطقي NOT لقيمة A



الشكل (١٨) عملية الجمع المنطقي AND للقيمتين A وB



الشكل (١٩) عملية الجمع الحسابي ADD للقيمتين A وB

نلاحظ من الأشكال السابقة أن كل عملية باستخدام المعالج الذي يحتوي ذاكرة مخبأة تستهلك زمناً مختلفاً عن العمليات الأخرى، وذلك بحسب الحالة الراهنة للذاكرة المخبأة، وينتج عن ذلك ضياعات زمنية بالإضافة إلى عدم قدرة على التنبؤ بالزمن الذي تحتاجه العملية، وذلك ماجعل هذا النوع من الذواكر غير مرغوب به في نظم CPS.

الاستنتاجات والتوصيات

قام بحثنا الحالي بتسليط الضوء على مكون جديد يمكن أن يحدث تطوراً كبيراً في نظم CPS الحديثة، والتي تشكل مجالاً لكثير من الأبحاث الحالية والمستقبلية. حيث وجدنا أنه من أهم الشروط الواجب توافرها في هذا النوع من الأنظمة هو الزمن والتنبؤ بزمن الاستجابة، وقد تبين أن ذاكرة SPM تحقق هذه الشروط بتأمين زمن ثابت ومحدد لكل عملية وواضحة بالنسبة إلى المبرمج، فكانت بهذه الميزات المكون المثالي الذي يحل مكان الذاكرة المخبأة في المعالجات الحديثة، لأن زمن الاستجابة للعملية في الذاكرة المخبأة يتعلق بالحالة الراهنة للذاكرة والمكونات الموجودة بها، بحيث يكون المبرمج غير قادر على تحديد حالة الذاكرة في لحظة زمنية معينة.

ومن هذه النتائج كان بناء نواة معالج قادر على التعامل مع نظم CPS، وبانتظار التطويرات الأخرى المستقبلية عليه مثل تعديل بقية مكونات العتاد الصلب وتقنيات الشبكات، وفتح المجال للتطويرات البرمجية وخاصة نظم التشغيل وخوارزميات الجدولة، والتي ستشكل مع بعضها المعالج الاحترافي الخاص بنظم CPS والذي سيلبي جميع متطلبات هذه النظم.

المراجع

[1].LEE,E.A ; *SESHIA,S.A. INTRODUCTION TO EMBEDDED SYSTEMS A CYBER-PHYSICAL SYSTEMS APPROACH*. 2nd .ed, MIT Press, Cambridge ,Massachusetts, USA, 2017, 585.

[2].SHRIVASTVA,A;DEALER,P; BABOUD,Y,L; STANTON,K; KHAYATIAN,M; ANDRADE,H,A; WEISS,M; EIDSON,J; CHANDHOKE,S. 2016, Time in Cyber-Physical Systems. National Institute of Standards and Technology USA.

[3].LANGGUTH,D.*2008 Scratchpad memory vs Caches Performance and Predictability comparison.*

[4].LEE, E ,A;EDWARDS,S,A.2007 *Precision Timed (PRET) Computation in Cyber-Physical Systems. Updated Position Paper for the National Workshop on High Confidence Software Platforms for Cyber-Physical Systems*: Research Needs and Roadmap November 30 - December 1, 2006, Alexandria, Virginia,USA.

[5].LEE,E,A. 2008, Cyber Physical Systems*: Design Challenges. Internatinal Symposium on Object/Component/Service-Oriented* Real_Time Distributed Computing(ISORC) USA,7.

[6].ROOB,J.2013 *An Introduction to the Research on Scratchpad Memory: Definition, Hardware,* Known Implementations and WCET Optimisation. University of Kaiserslautern, Embedded Systems Group Germany,19.

[7].LEE,E,A.2015, *The Past, Present and Future of Cyber-Physical Systems: A Focus on Models.* Sensors USA, 15, 4837-4869.

[8].WASLY,S . 2018, *Scratchpad Memory Management For Multicore Real-*Time Embedded Systems. A thesis presented to the University of Waterloo in fulfillment of the thesis requirement for the degree of Doctor of Philosophy in Electrical and Computer Engineering Canada.

[9].LAPLANTE ,P, A.2004. *Real-Time Systems Design and Analysis. IEEE Press & Wiley Inter science* United States of America, third edit edition.

[10].Scratchpad memory. http://en.wikipedia.org/wiki/Scratchpad_memory.

[11].STALLINGS,W. 2013. *Computer Organization And Architecture Designing For Performance. PEARSON* United States of America, 135_182.

[12].MIKKO,P; LIPASTI.H. 2016, *Cache Replacement Policies*. University of Wisconsin-Madison United States.

[13].WASLY,S. 2012, *A Dynamic Scratchpad Memory Unit for Predictable Real-Time Embedded Systems*. A thesis presented to the University of Waterloo in fulfillment of the thesis requirement for the degree of Master of Applied Science in Electrical and Computer Engineering, Waterloo, Ontario, Canada.

[14].CHANDRAKASAN,A; RABAEY,J,M. 2002, *Digital Integrated Circuits. Prentice-Hall*.

[15].BANAKAR,R; STEIKE,S; LEE,B,B; BALAKRISHNSN,M; ARWEDEL,P. 2002 *Scratchpad memory: design alternative for cache on-chip* memory in embedded systems. ACM Press, In Proceedings of the . . . , page 73, New York, New York, USA.

[16].Altera. https://en.wikipedia.org/wiki/**Altera**.

[17].ALBUSTANI,H.2013, *Digital Circuits And Systems Modeling*. Tishreen University, Syria.

[18].ModelSim. https://en.wikipedia.org/wiki/**ModelSim**.