مجلة جامعة تشرين للبحوث والدراسات العلمية _ سلسلة العلوم الهندسية المجلد (40) العدد (2) 2018
Tishreen University Journal for Research and Scientific Studies - Engineering Sciences Series Vol. (40) No. (2) 2018

Analytical Study on Scheduling Algorithms of Periodic Tasks in Real Time Systems on Multicore Processor

Dr. Mohammed Hijazieh *
Khaldon Faqi **

(Received 5 / 1 / 2018. Accepted 1 / 3 / 2018)

□ ABSTRACT □

Real Time Systems are considered nowadays as the most common systems, because its wide spread and usage in many areas including technical and applied researches, also the installation of such systems on multicore platforms made them very desirable as embedded systems and control units, because of its high performance and robustness comparing to multiprocessors platforms.

Scheduling is the basic operation in real time systems, and relies in ordering the execution of tasks depending on priorities witch is set according to scheduling policies.

This paper aims to introduce an analytical study of the most important scheduling algorithms to find the one with best performance according to number of parameters such as system load, context switching overheads, and scheduling overheads, when they applied to number of periodic tasks generated randomly.

We use for this purpose SIMSO real time systems simulator, because of its reliability, robustness, and the support to a large number of scheduling algorithms, and cache memory simulation with its different levels witch is considered to be the main component in multicore platforms.

Key Words: Scheduling, Periodic Tasks, Multicore Processor.

*Associate Professor, Department of computer and automatic control Engineering, Faculty of Mechanical and electrical Engineering, Tishreen University, Lattakia, Syria.
**Postgraduate Student , Department of computer and automatic control Engineering, Faculty of Mechanical and electrical Engineering, Tishreen University, Lattakia, Syria.

مجلة جامعة تشرين للبحوث والدراسات العلمية _ سلسلة العلوم الهندسية المجلد (40) العدد (2) 2018
Tishreen University Journal for Research and Scientific Studies - Engineering Sciences Series Vol. (40) No. (2) 2018

دراسة تحليلية لخوارزميات جدولة المهام الدورية في نظم الزمن الحقيقي على معالج متعدد النوى

الدكتور محمد حجازية*
خلدون فاقي**

(تاريخ الإيداع 5 / 1 / 2018. قُبِل للنشر في 1/ 3 / 2018)

□ ملخّص □

تعتبر نظم الزمن الحقيقي من أكثر النظم شيوعاً في الوقت الحاضر، نظراً لانتشارها الواسع واستخدامها في شتى المجالات بما في ذلك الأبحاث التقنية والتطبيقية، فضلاً عن أنّ تطبيق مثل هذه النظم على منصات عمل متعددة النوى أضحى أمراً مرغوباً كما هو الحال في الأنظمة المضمنة ووحدات التحكم بسبب سرعة أداء مثل هذه المنصات وتماسكها مقارنة مع المنصات الأخرى متعددة المعالجات والتي تعاني من بطء في تبادل المعطيات المختلفة بسبب مقدار سرعة قنوات الاتصال بينها والذي عادة ما يكون أبطأ من ذلك الموجود في المنصات متعددة النوى.

تشكل جدولة المهام محور عمل نظم الزمن الحقيقي، وهي في حقيقتها تقوم على مبدأ ترتيب تنفيذ المهام اعتماداً على الأفضليات المسندة لها، وتختلف عملية الإسناد هذه باختلاف الخوارزمية المتبعة في الجدولة.

يهدف هذا البحث إلى تقديم دراسة تحليلية لأهم خوارزميات جدولة المهام الدورية وذلك لمقارنة أدائها على منصة متعددة النوى من حيث مجموعة من البارامترات مثل حمل المعالج، أعباء عمليات تبديل السياق، وأعباء اتخاذ قرار الجدولة، وذلك من أجل انتخاب الخوارزمية الأفضل من بين هذه الخوارزميات من حيث البارامترات المعتبرة عند تطبيقها على مجموعة من المهام الدورية المولدة عشوائياً.

تم استخدام المحاكي SIMSO لهذا الغرض، وذلك بسبب موثوقيته ودعمه لعدد كبير من خوارزميات الجدولة، بالإضافة إلى أنه يحاكي استخدام ذاكرة الكاش بمستوياتها المختلفة والتي تعتبر حجر الأساس في بنية المنصات متعددة النوى.

الكلمات المفتاحية: الجدولة، المهام الدورية، المعالج متعدد النوى.

* **أستاذ مساعد – قسم هندسة الحاسبات والتحكم الآلي– كلية الهندسة الميكانيكية والكهربائية– جامعة تشرين– اللاذقية– سورية.**
** **طالب دراسات عليا (دكتوراه) – قسم هندسة الحاسبات والتحكم الآلي– كلية الهندسة الميكانيكية والكهربائية– جامعة تشرين– اللاذقية– سورية.**

مقدمة:

نظم الزمن الحقيقي هي النظم التي تعتمد صحة التنفيذ فيها ليس على سلامة النتيجة المنطقية للعملية فحسب وإنما على الزمن الذي نحصل فيه على النتيجة، ونسمي هذا القيد الزمني Deadline فإذا حصلنا على النتيجة المنطقية السليمة لتنفيذ العملية بعد تجاوز القيد الزمني فإنه من الممكن أن نحصل على نتائج كارثية وذلك تبعاً لنوع النظام.

تصنف نظم الزمن الحقيقي بشكل عام إلى نظم قاسية Hard Real Time Systems ونظم لينة Soft Real Time Systems.

يؤدي تجاوز القيد الزمني في النظم القاسية إلى نتائج كارثية قد ينتج عنها انهيار النظام بكليته أو جزء منه، بينما في النظم اللينة لا يحدث هذا الأمر أي يمكن التساهل مع التجاوزات التي قد تحدث على القيد الزمني.

أهمية البحث وأهدافه:

يهدف هذا البحث إلى تقديم دراسة تحليلية شاملة من أجل تقييم أداء بعض خوارزميات الجدولة الأكثر أهمية في جدولة المهام الدورية Periodic Tasks، بهدف الوصول إلى انتخاب إحدى هذه الخوارزميات لكي تكون الأفضل من بين مجموعة الخوارزميات المستخدمة.

تتم مقارنة الخوارزميات وفقاً لمجموعة من البارامترات المختارة والتي تلعب دوراً أساسياً في تقييم أداء مثل هذه الخوارزميات في نظم الزمن الحقيقي، وهذه البارامترات هي: حمل النظام System Load، وأعباء عمليات تبديل السياق Context Switch Overheads، وأعباء الجدولة Scheduling Overheads.

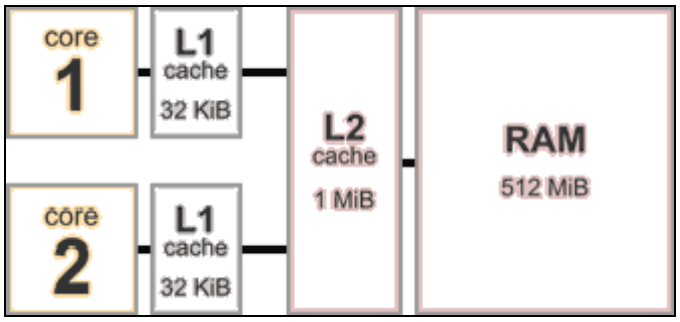
تطبق هذه الخوارزميات على مجموعة من المهام الدورية المولدة بشكل عشوائي مع الأخذ بعين الاعتبار كل العوامل المؤثرة مثل حجم ذاكرة الكاش Cache Memory وعدد مستوياتها، وأزمنة التأخير الناتجة عن الوصول إلى ذاكرة الكاش أو إلى الذاكرة الرئيسية RAM، وذلك لكون المنصة المستخدمة هي عبارة عن معالج متعدد النوى Multicore Processor.

طرائق البحث ومواده:

نستعرض فيما بعض التفاصيل حول طبيعة المهام وبنية المعالج متعدد النوى، وهيكلية ذاكرة الكاش.

1. بنية المعالج متعدد النوى Multicore Processor Architecture:

يتألف المعالج متعدد النوى من مجموعة من النوى Cores المتوضعة على شريحة واحدة Single Chip، ويكون لكل نواة ذاكرة كاش خاصة بها وهي ما نسميها بالمستوى الأول (L1) First Level Cache، وتشترك هذه النوى فيما بينها بذاكرة كاش أخرى أكبر حجماً لكن أقل سرعة من المستوى الأول وهذا ما نسميه بالمستوى الثاني (L2) Second Level Cache، والذي بدوره يكون متصلاً مع الذاكرة الرئيسية RAM، ويبين الشكل (1) بنية المعالج متعدد النوى.[6][2]



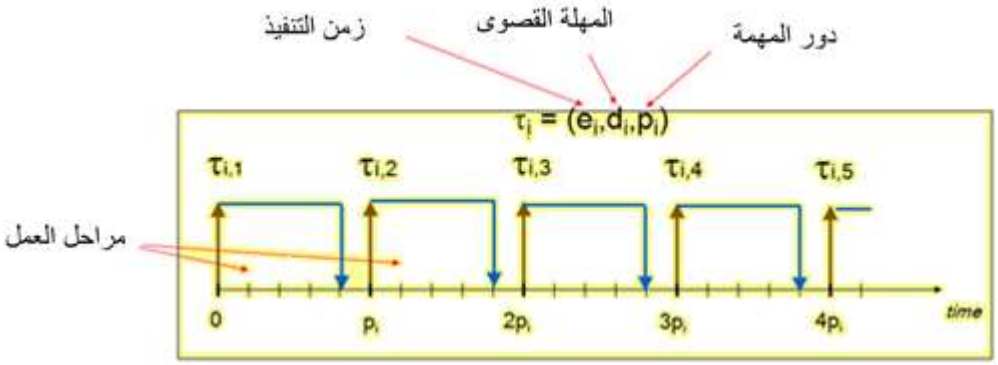
الشكل (1) يبين بنية المعالج متعدد النوى

2. المهام في نظم الزمن الحقيقي Tasks of Real Time systems:

تصنف المهام في نظم الزمن الحقيقي بشكل عام إلى مهام دورية Periodic Tasks، ومهام عشوائية Sporadic Tasks، ومهام غير دورية Aperiodic Tasks.[4]

ويكون لكل مهمة ثلاثة بارامترات هي زمن التنفيذ Execution Time، والفترة الفاصلة بين إصدارين متتاليين للمهمة Inter Arrival Times والتي تكون قيمتها أكبر من الصفر ومتساوية في المهام الدورية، بينما تكون أكبر من الصفر ومتغيرة في المهام العشوائية، أما في المهام غير الدورية فتكون متغيرة وقد تساوي الصفر.

أما البارامتر الثالث فهو القيد الزمني Deadline. ويبين الشكل (2) الشكل العام للمهمة في نظم الزمن الحقيقي.



الشكل (2) يبين الشكل العام للمهمة في نظم الزمن الحقيقي

هناك مجموعة من البارامترات التي تخص كل مهمة، ولا بد من إعطاء هذه البارامترات قيم معينة حتى نتمكن من محاكاة هذه المهام على المعالج متعدد النوى، وهذ البارامترات هي:[11]

1. زمن التفعيل Activation Time: هو الزمن الذي يبدأ عنده إطلاق أول مرحلة عمل job للمهمة الحالية.
2. زمن تنفيذ الحالة الأسوأ (Worst Case Execution Time) WCET: هو أكثر زمن يمكن للمهمة الحالية أن تستغرقه خلال تنفيذها على المعالج بغض النظر عن أزمنة الانقطاع الناتجة عن استبعاد المهمة من التنفيذ نتيجة لورود مهمة ذات أفضلية أعلى.
3. دور المهمة Period: هو الفترة الزمنية الفاصلة بين كل إطلاقين متتاليين للمهمة؛ أي بين كل مرحلتي عمل Two Jobs.

4. القيد الزمني Deadline: هو الفترة الزمنية التي يتوجب على المهمة أن تنهي تنفيذها خلاله، وإلا فقد نحصل على فشل في النظام.

5. عدد دورات الساعة لكل تعليمة (Clock Per Instruction) CPI: هي عدد دورات الساعة اللازمة لتنفيذ كل تعليمة من التعليمات التي تتضمنها المهمة؛ حيث أن المهمة ممكن أن تتضمن أكثر من تعليمة في بعض الأحيان.

6. عدد التعليمات Instructions: عدد التعليمات التي تتضمنها المهمة.

7. معدل الوصول إلى الذاكرة Memory Access Rate: تمثل النسبة بين عدد التعليمات التي تتطلب الوصول إلى الذاكرة بالنسبة إلى عدد التعليمات الكلي، ويسمى بـ (MIX).

8. هيئة مسافة المكدس (Stack Distance Profile) SDP: وتتضمن توزع محتويات الكاش المختلفة والتي تسمى بـ (Cache Lines) ضمن ذاكرة الكاش وتقاس عن طريق عدد الـ Cache Lines المنفردة Unique التي تفصل بين وصولين متتاليين لنفس الـ Cache Line.

برنامج المحاكاة المستخدم في البحث:

تم استخدام المحاكي SIMSO في هذا البحث لكونه يتمتع بالخصائص التالية:

1. مفتوح المصدر Open Source: أي يمكن تعديل أي ملف من المكتبات المضمنة في هذا المحاكي بما في ذلك الملفات التي تصف عمل خوارزميات الجدولة.

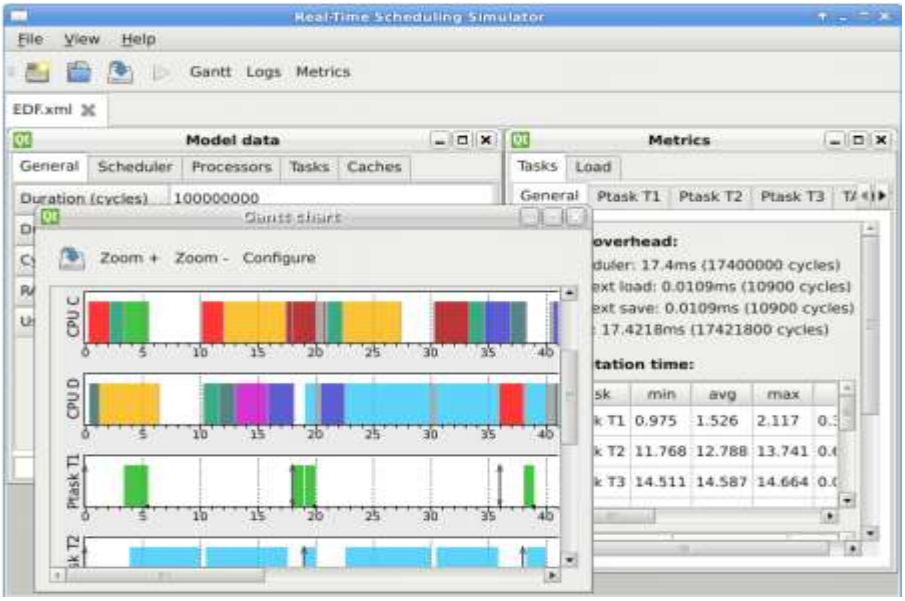
2. يؤمن واجهة مستخدم رسومية (Graphical User Interface) GUI: وذلك من خلال مجموعة من الأزرار والخيارات التي تتيح ضبط إعدادات المحاكاة بشكل سهل ومرن.

3. يؤمن إحصائيات المحاكاة في الخرج بعد الانتهاء من عملية المحاكاة وهذه الإحصائيات تتضمن مثلاً أزمنة إطلاق المهمة، وكلفة عمليات تبديل السياق، وكلفة عمليات اتخاذ قرار الجدولة، وغيرها من الإحصائيات المفيدة.

4. مكتوب بلغة الـ Python والتي تعد من اللغات البرمجية المتطورة، كما أنها تدعم البرمجة غرضية التوجه (Object Oriented Programming) OOP.

5. السرعة في التنفيذ مقارنة مع محاكيات أخرى ضمن نفس المجال.

ويبين الشكل (3) الواجهة الرسومية للمحاكي SIMSO.



الشكل (3) يبين الواجهة الرسومية للمحاكي SIMSO

النتائج والمناقشة:

تمت دراسة ثلاثة سيناريوهات مختلفة تتضمن العمل وفق ثلاث خوارزميات هي:

1. خوارزمية (Rate Monotonic) RM.
2. خوارزمية (Earliest Deadline First) EDF.
3. خوارزمية (Largest Local Remaining Execution time First) LLREF.

يتم في السيناريو الأول تنفيذ المحاكاة على نواتين، وفي السيناريو الثاني على أربع أنوية، وفي السيناريو الثالث على ثمان أنوية. مع الأخذ بعين الاعتبار البارامترات التالية:

1. كل نواة تحتوي على ذاكرة L1 كاش.
2. L2 كاش مشتركة بين جميع الأنوية.
3. كلفة الوصول إلى الـ RAM هي (100 cycles).
4. كلفة تبديل السياق (100 cycles).

ويبين الجدول (1) قيم جميع البارامترات المتعلقة بذاكرة الكاش باختلاف مستوياتها.

الجدول (1) قيم بارامترات ذاكرة الكاش

نوع الكاش Cache Type	الحجم Size	زمن الوصول Access Time	كلفة الضياع Miss Penalty	حجم الـ Cache Line
L1	4 KB	1 cycle	9 cycle	64 byte
L2	64 KB	10 cycle	90 cycle	64 byte

كما يبين الجدول (2) مجموعة المهام الدورية المولدة بشكل عشوائي مع قيم البارامترات الخاصة بها.

الجدول (2) يبين المهام الدورية

المهمة Task	زمن التنفيذ WCET	الدور Period	القيد الزمني Deadline	عدد التعليمات Instructions Count	نسبة التعليمات الذاكرية MIX
T1	40	50	50	1300000	0.8
T2	9	20	20	600000	0.8
T3	7	15	15	500000	0.8
T4	5	15	15	500000	0.8
T5	2	10	10	300000	0.7
T6	6	10	10	500000	0.5
T7	4	10	10	500000	0.4
T8	3	10	10	300000	0.5

تمت مقارنة النتائج بالاعتماد على البارامترات التالية:

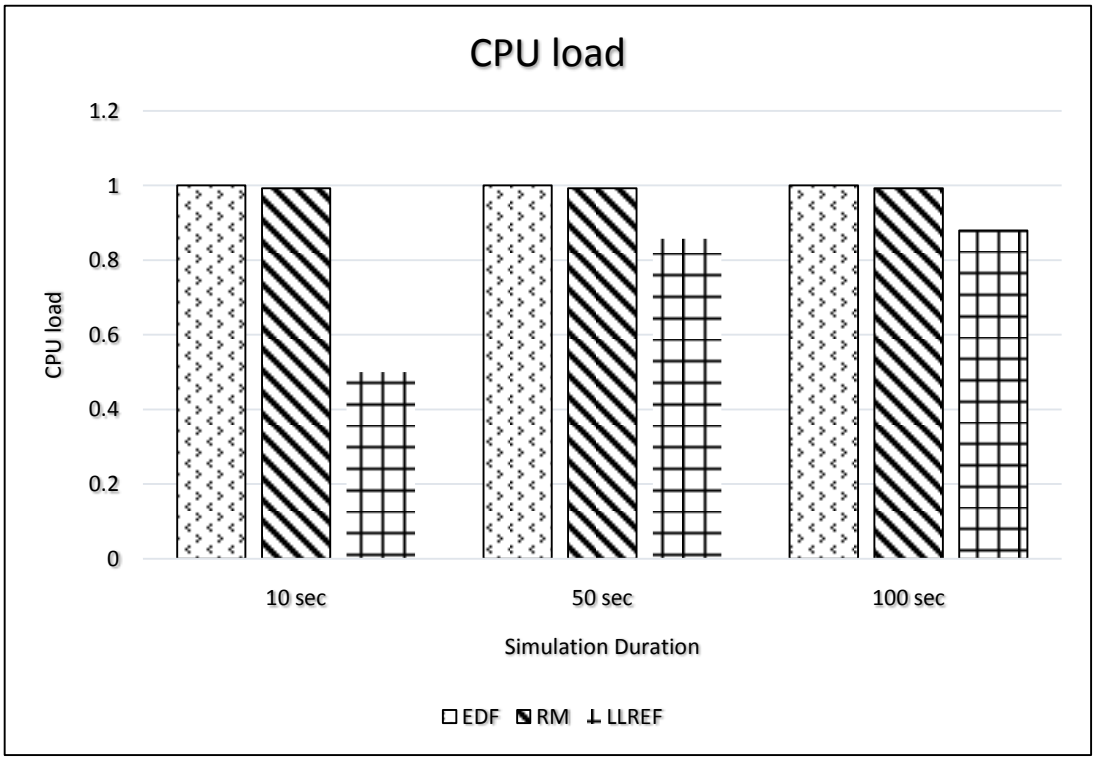
1. حمل المعالج CPU Load: ويمثل المقدار الزمني الذي يكون فيه المعالج مشغول بالتنفيذ بالنسبة إلى الزمن الكلي للمحاكاة.

2. أعباء عمليات تبديل السياق Context Switching Overheads: وتتضمن الزمن المنقضي ريثما يتم تبديل مهمة تنفذ حالياً بمهمة أخرى ذات أفضلية أعلى من المهمة الحالية وما يترتب على ذلك من حفظ حالة المهمة المستبعدة وتحميل مسجلات المعالج بالقيم الجديدة للمهمة الأخرى.

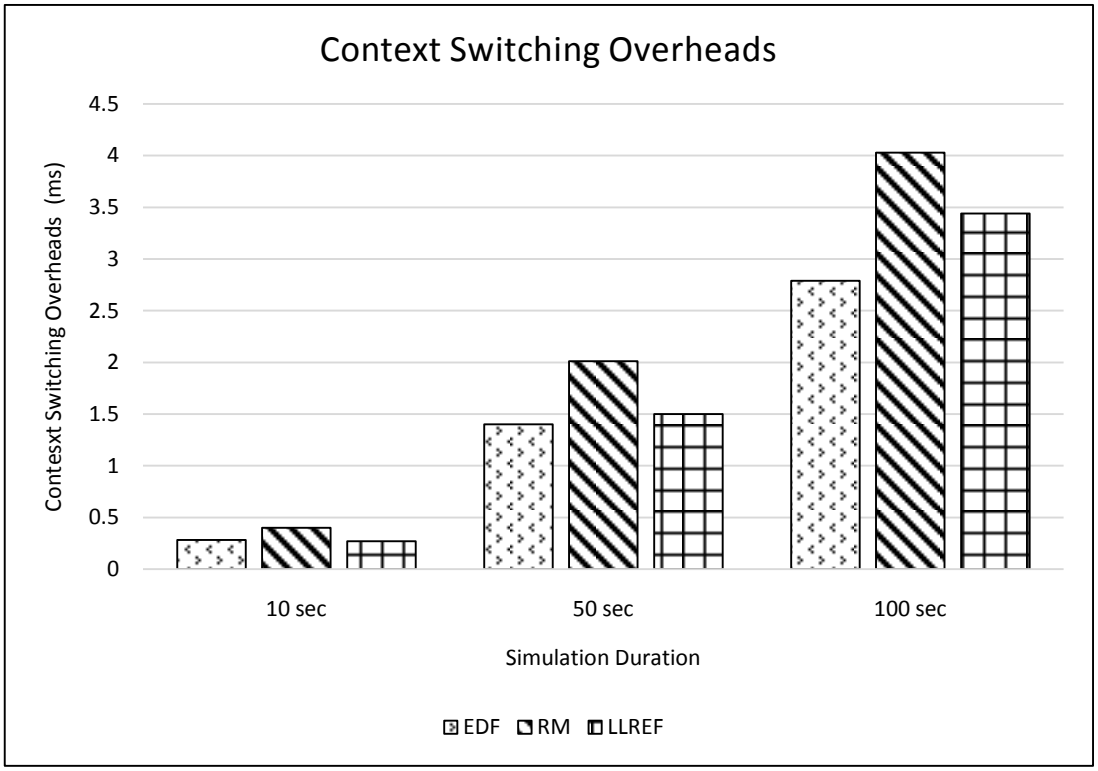
3. أعباء الجدولة Scheduling Overheads: وتتضمن الزمن المستهلك من قبل المعالج عند ورود حدث جديد يستدعي نداء المجدول لكي يقوم باتخاذ قرار الجدولة وما يترتب على ذلك من تنفيذ العالج لبعض العمليات حتى يتمكن من اتخاذ قرار الجدولة.

1. السيناريو الأول:

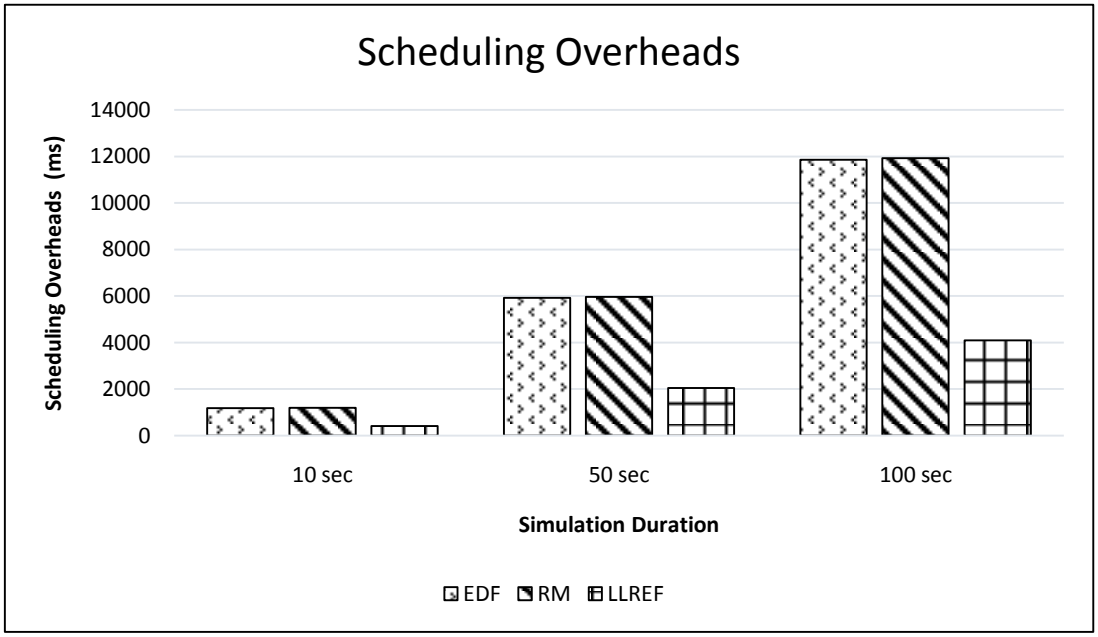
تم في هذا السيناريو محاكاة تطبيق خوارزميات الجدولة الثلاثة المذكورة سابقاً على مجموعة من المهام الدورية وتبين الأشكال (4)، (5)، (6) المقارنة بين خوارزميات الجدولة الثلاثة من ناحية حمل المعالج، وأعباء عمليات تبديل السياق، وأعباء الجدولة، وذلك عند العمل على نواتين.



الشكل (4) المقارنة من حيث حمل المعالج.



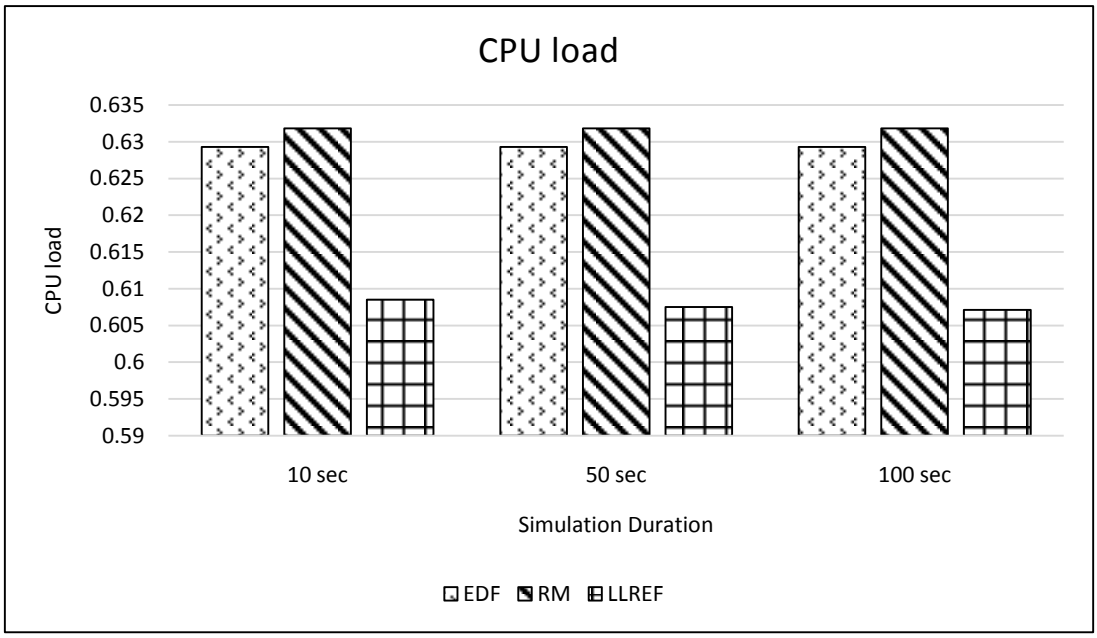
الشكل (5) يبين المقارنة من حيث أعباء تبديل السياق



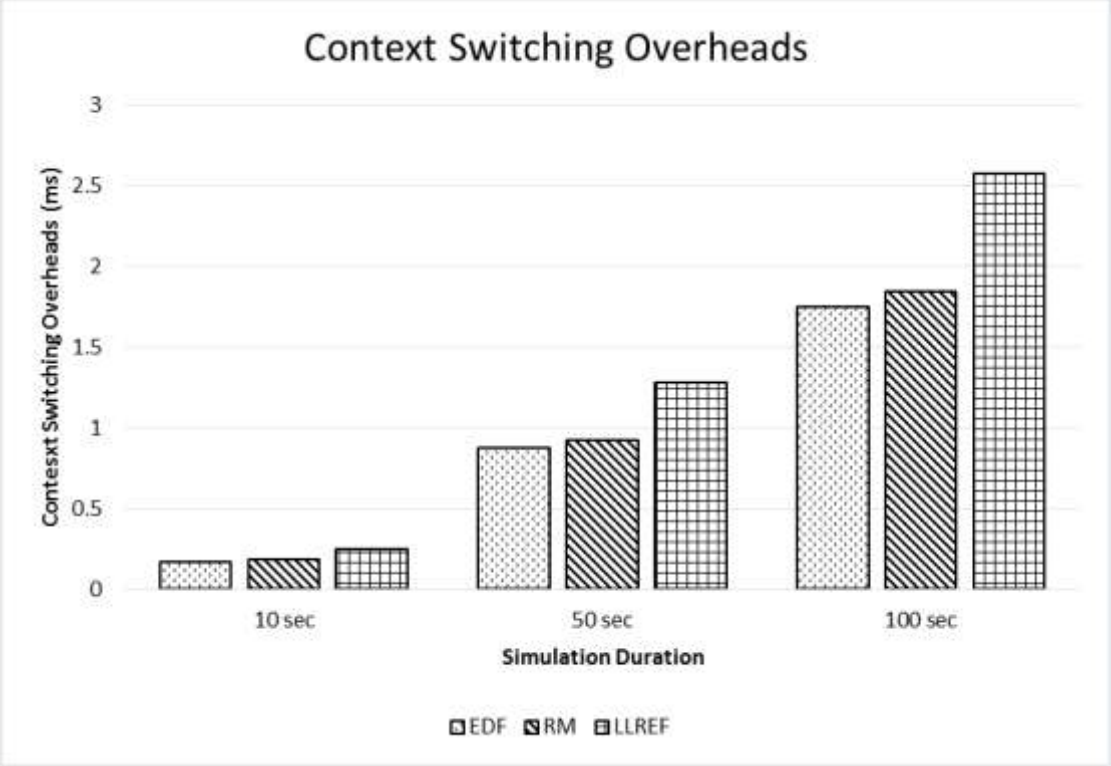
الشكل (6) يبين المقارنة بين أعباء الجدولة

2. السيناريو الثاني:

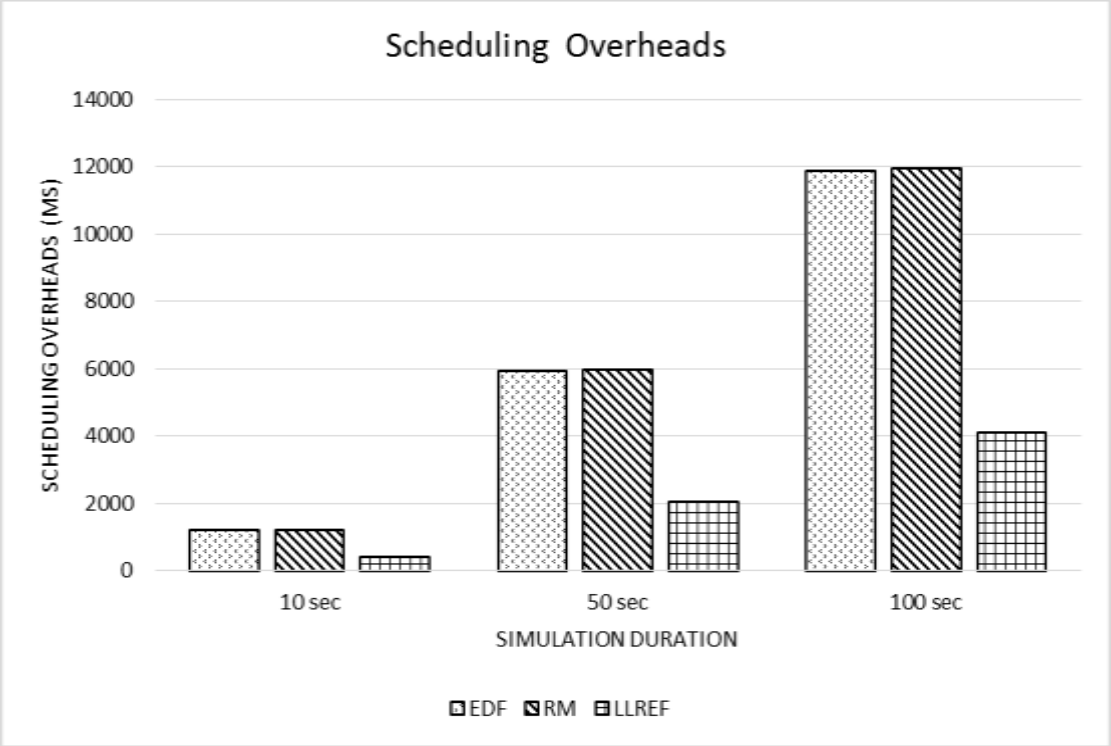
يتضمن السيناريو الثاني نفس الخطوات التي تمت في السيناريو السابق لكن مع وجود أربع أنوية، وتبين الأشكال (7)، (8)، (9) المقارنة بين خوارزميات الجدولة الثلاثة من ناحية حمل المعالج، وأعباء عمليات تبديل السياق، وأعباء الجدولة، وذلك عند العمل على أربع أنوية.



الشكل (7) يبين المقارنة من حيث حمل المعالج



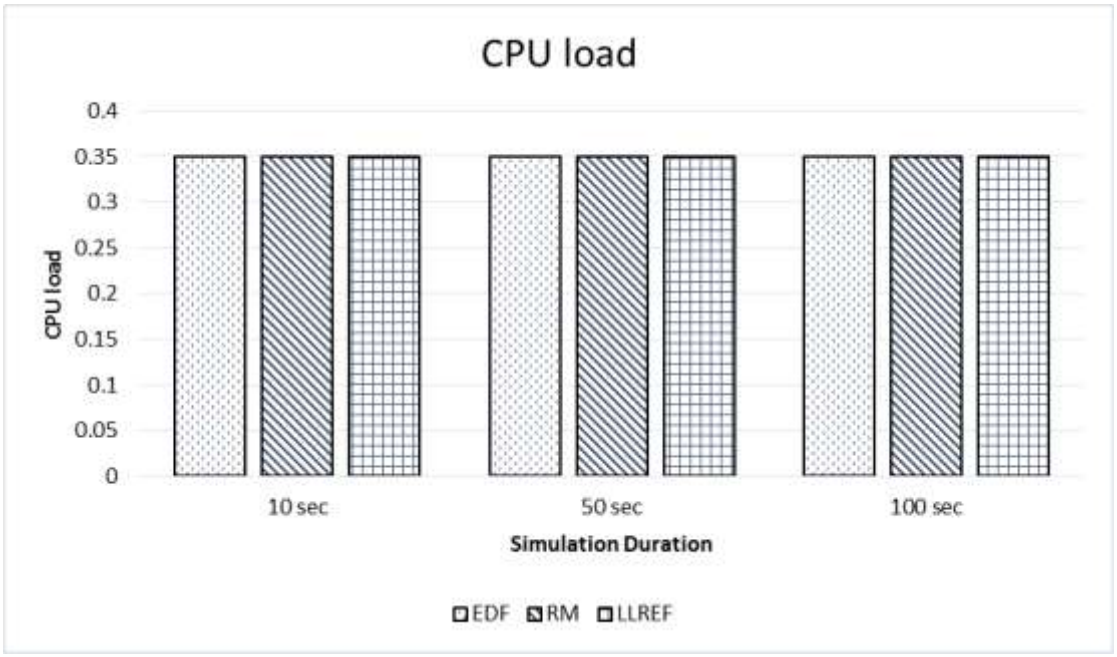
الشكل (8) يبين المقارنة من حيث أعباء تبديل السياق



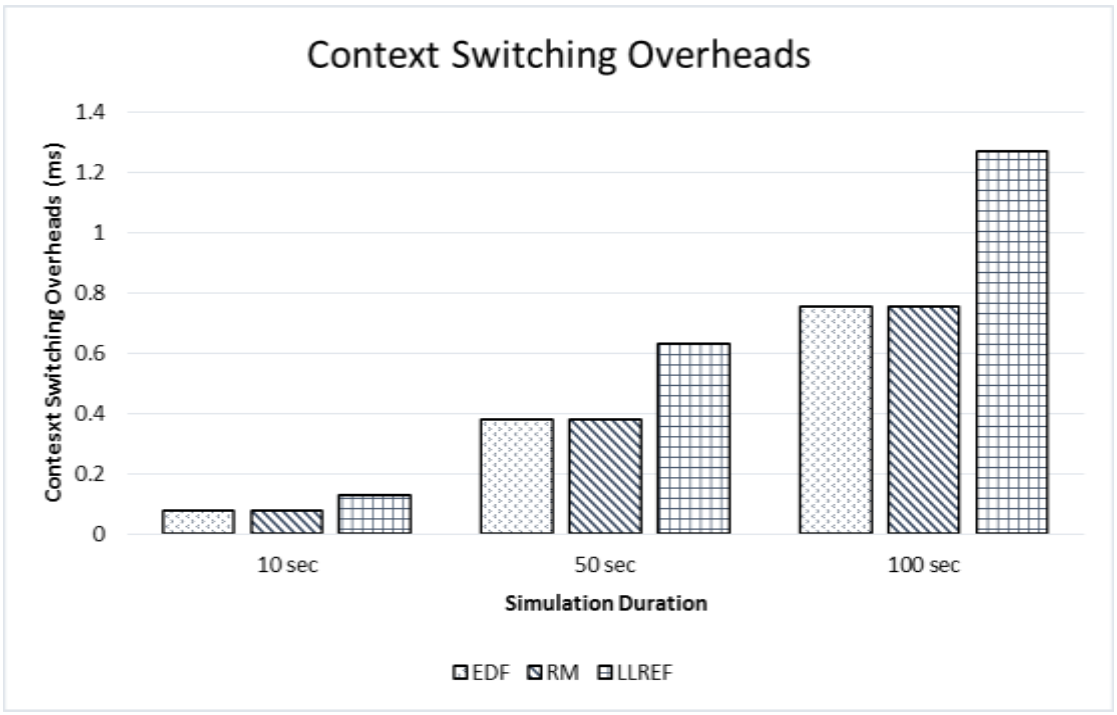
الشكل (9) يبين المقارنة من حيث أعباء الجدولة

3. السيناريو الثالث:

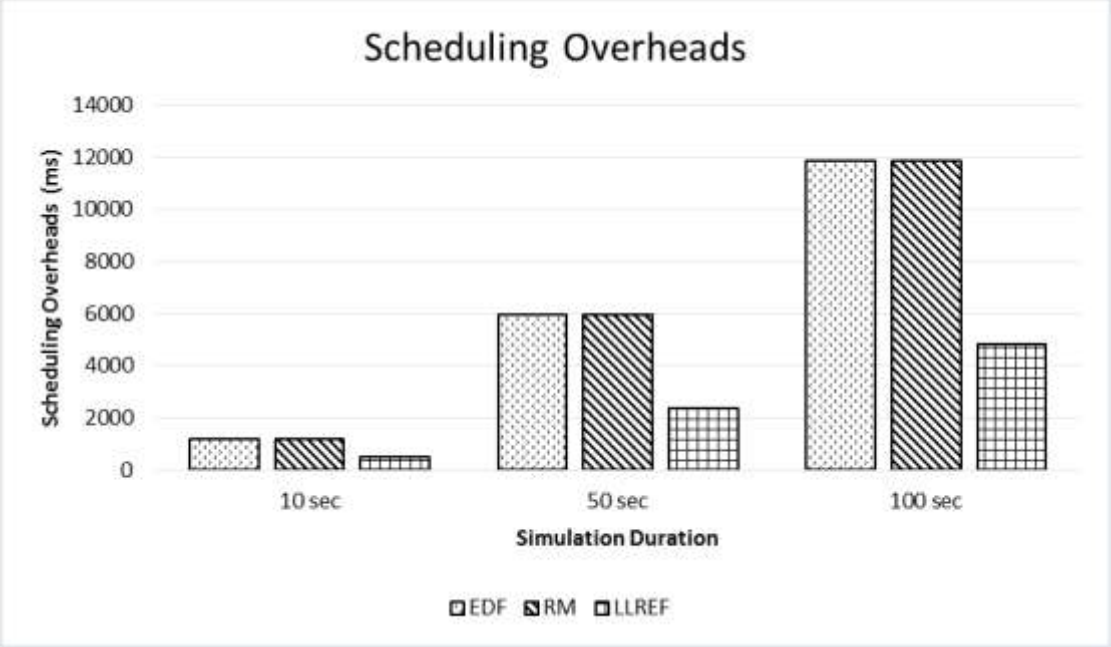
يتضمن السيناريو الثالث نفس الخطوات التي تمت في السيناريو السابق لكن مع وجود ثمان أنوية، وتبين الأشكال (10)، (11)، (12) المقارنة بين خوارزميات الجدولة الثلاثة من ناحية حمل المعالج، وأعباء عمليات تبديل السياق، وأعباء الجدولة، وذلك عند العمل على ثمان أنوية.



الشكل (10) يبين المقارنة من حيث حمل المعالج



الشكل (11) يبين المقارنة من حيث أعباء الجدولة



الشكل (12) يبين المقارنة من حيث أعباء الجدولة

الاستنتاجات والتوصيات:

الاستنتاجات:

يمكن من خلال ملاحظة نتائج المقارنات بين قيم البارامترات المأخوذة بعين الاعتبار في السيناريوهات الثلاثة أنّ:

1. أعباء الجدولة تكون كبيرة نسبياً عند العمل على معالجين وتقل هذه الأعباء كلما زاد عدد المعالجات، بسبب أنّ زيادة عدد المعالجات يخفف من العمليات اللازمة لاتخاذ قرار الجدولة.

2. أعباء العمليات الناتجة عن تبديل السياق تكون كبيرة في السيناريو الأول ثم تقل في السيناريوهين الثاني والثالث بسبب توفر عدد أكبر من المعالجات، وبالتالي لا داعي لاستبعاد المهمة التي تنفذ حالياً عند ورود مهمة أخرى ذات أفضلية أعلى.

3. حمل المعالج يكون كبيراً في البداية لكن مع ازدياد عدد المعالجات يبدأ هذا الحمل بالتناقص بسبب زيادة الوقت الشاغر للمعالج.

4. خوارزمية LLREF تعتبر أفضل من الخوارزميات الأخرى من حيث حمل المعالج وأعباء الجدولة، في حين أن خوارزمية EDF تعطينا أداء أفضل من ناحية أعباء تبديل السياق.

التوصيات:

يمكن بعد الاطلاع على الاستنتاجات السابقة العمل على البحث عن طريقة تقلل من أعباء تبديل السياق في خوارزمية LLREF كون أن هذه الخوارزمية قدمت أفضل أداء بين الخوارزميات الأخرى من حيث البارامترات المتعلقة بحمل المعالج وأعباء الجدولة، كما يوصي البحث بتكرار السيناريوهات الثلاثة السابقة على عدد أكبر من الخوارزميات بغية التعرف على أدائها، بالإضافة إلى زيادة زمن المحاكاة ومقارنة النتائج مع تلك التي حصلنا عليها في السيناريوهات المستعرضة في هذا البحث.

المراجع:

1. ANKUR JAIN , **"** *Multishare Task Scheduling Algorithm For Real Time Micro-controller Based Application* **" ,** Mechatronics and Applications: An International Journal (MECHATROJ), Vol. 1, No.1, 2015.

2. CHARU RANI, MRS. MANJU GODARA , **"** *Real Time System Scheduling Algorithms & Fault Tolerance* **"** , International Journal of Advanced Research in Computer and Communication Engineering , Vol. 4, Issue 7, July 2015.

3. SRI RAJ PRADHAN, SITAL SHARMA, DEBANJAN KONAR, **"***A Comparative Study on Dynamic Scheduling of Real-Time Tasks in Multiprocessor System using Genetic Algorithms "* , International Journal of Computer Applications , Volume 120 – No.20, June 2015.

4. CHRISTINE ROCHANGE,**"***Parallel Real-Time Tasks, as Viewed by WCET Analysis and Task Scheduling Approaches***"**, University of Toulouse, Toulouse, France 2016.

5. FERNANDA F. PERONAGLIO AND ALEARDO MANACERO, "*Modeling Real-Time Schedulers For Use In Simulations Through A Graphical Interface*", Dept of Computer Science and Statistics São Paulo State University – UNESP, 2017.

6. KARTHIK S. LAKSHMANAN, **"***Scheduling and synchronization for Multicore Real Time Systems "***,** Carnegie Mellon University, Pittsburgh, PA, 2011.

7. JOHN CARPENTER, SHELBY FUNK, PHILIP HOLMAN, ANAND SRINIVASAN,JAMES ANDERSON, AND SANJOY BARUAH , "*A Categorization of Real-time Multiprocessor Scheduling Problems and Algorithms"* , Department of Computer Science, University of North Carolina at Chapel Hill, 2010.

8. MANAR QAMHEIH, **"***Scheduling of Parallel Real-time DAG Tasks on Multiprocessor Systems***",** University of Paris-est 2015.

9. DAN MCNULTY, LENA OLSON, MARKUS PELOQUIN , "*A Comparison of Scheduling Algorithms for Multiprocessors*" , University of Maryland ,2010.

10. GREG LEVIN , "*DP-FAIR: A Simple Model for Understanding Optimal Multiprocessor Scheduling"* International Journal of Computer Applications (0975 – 8887) Volume 110 – No. 6 , 2009.

11. KUSHAL ANJARIA, ARUN MISHRA, "*Thread scheduling using ant colony optimization: An intelligent scheduling approach towards minimal information leakage"*, Department of Computer Science & Engineering, DIAT, Pune, India, 2017.

12. GOKUL SIDARTH THIRUNAVUKKARASU*, AND RAGIL KRISHNA, "*Scheduling Algorithm for Real-Time Embedded Control Systems Using Arduino Board",* Deakin University, School of Engineering, Waurn ponds, Australia, 2017.

13. AMJAD MAHMOOD , SALMAN A. KHAN, "*Energy-Aware Real-Time Task Scheduling in Multiprocessor Systems Using a Hybrid Genetic Algorithm*", University of Bahrain, 2017.

14. LINLIN TANG*A*, KAIQIANG MA, ZUOHUA LI, "*A New Scheduling Algorithm Based on Ant Colony Algorithm and Cloud Load Balancing*", Harbin Institute of Technology Shenzhen Graduate School Shenzhen, China, 2016.

15. JAYARAM MUDIGONDA, "*Fast Switching of Threads Between Cores*", ACM SIGOPS Operating Systems Review special issue, 2009.

16. SISU XI, MENG XU, CHENYANG LU,' *Real-Time Multi-Core Virtual Machine Scheduling in Xen',* Washington University, 2013.

17. MENG XU LINH T.X. PHAN INSUP LEE OLEG SOKOLSKY, "*Cache-Aware Compositional Analysis of Real-Time Multicore Virtualization Platforms", University of Pennsylvania*, 2013.

18. VIJAYSHREE SHINDE, "*Comparison of Real Time Task Scheduling Algorithms"*, Terna Engineering College, Navi Mumbai, India, 2017.

19. GIRISH S. THAKARE, DR. PRASHANT. R. DESHMUKH, "*Performance Analysis of Real Time Task Scheduling Algorithm"*, Maharashtra, India ,2017.

20. ANKUR JAIN, "*Multishape Task Scheduling Algorithm For Real Time Micro-Controller Based Application*", Department of Computer Engineering, ABV-IIITM, Gwalior ,2017.

21. JALIL BOUKHOBZA, "*Cache Memory Aware Priority Assignment and Scheduling Simulation of Real-Time Embedded Systems*", UNIVERSITÉ DE BRETAGNE OCCIDENTALE, 2017.