النظم المطمورة: خصائصها وتصميمها ومعوليتها

محمود علي محمد رشدي[1] وعلي محمد علي رشدي[2]

[1] قسم هندسة الإلكترونيات والاتصالات الكهربائية، كلية الهندسة، جامعة القاهرة، الجيزة، جمهورية مصر العربية
mahmoud.rushdi@gmail.com

[2] قسم الهندسة الكهربائية وهندسة الحاسبات، كلية الهندسة، جامعة الملك عبد العزيز، ص. ب. 21598، جدة، 80204، المملكة العربية السعودية
arushdi@kau.edu.sa

المستخلص. تقدم هذه الورقة مراجعة واستعراضا لبعض ما تحفل به أدبيات النظم المطمورة، وتشرح من خلال ذلك كثيرا من المفاهيم الهامة المتعلقة بتصميم النظم الهندسية بوجه عام. يتم سرد الخصائص المميزة للنظم المطمورة مع إبراز خصائص تكريسها وتداخل مادياتها وبرمجياتها وتقيد عملها بالزمن الحقيقي. يجري وصف عملية تصميم النظم المطمورة، وبيان أنها لا تندرج تحت تصميم الماديات وحدها ولا تصميم البرمجيات وحدها، ولكنها عملية تصميم مشترك متزامن لهما، كما يتم شرح الفجوة القائمة بين هندسة النظم الحرجة وهندسة الجهد الأفضل، وتأثير هذه الفجوة على عملية تصميم النظم المطمورة. يتم استعراض تاريخ التقدم المستمر في تقانة الدوائر المكاملة لأشباه الموصلات واستهداف هذا التقدم لوضع نظام على رقاقة، كما تتم الإشارة إلى انتقال النموذج الإرشادي من أسلوب تصميم الدوائر المكاملة محددة التطبيق إلى أسلوب التصميم المبني على منصة، وهو أسلوب دخل عالم الدوائر المكاملة مؤخرا بعد أن حقق نجاحات باهرة في ميادين هندسية عديدة أخرى. كذلك يتم سرد أهم المجروحيات التي تعاني منها النظم المطمورة، وبيان كيف تتم الدراسة الكمية لهذه المجروحيات باستخدام شجرة الأخطاء، وإيضاح كيفية حساب وتحسين معولية النظم المطمورة. تختتم الورقة باقتراح تطوير خوارزميات معولية النظم الوافرة جزئيا للتنازل عن فرضية الاستقلال الإحصائي بين المكونات، ومن ثم جعل هذه الخوارزميات صالحة لنمذجة معولية النظم المطمورة. كذلك تقترح الورقة استكشاف التطبيقات المحتملة للهندسة العكسية في مجال النظم المطمورة.

الكلمات الدالة: النظم المطمورة، التصميم المشترك، العتاد المادي (الصلب)، العتاد البرمجي (الرخو)، العتاد الوسيط (المتماسك)، النظم الحرجة، النظام على رقاقة، المجروحية، المعولية.

1. مقدمة

النظام المطمور هو نظام محسابي متخصص الغرض (special-purpose computer system) مصمم لدعم ومساندة نظام تقاني أكبر يعرف بالنظام الطامِر (المُبْطِن أو الحاوي) (embedding system)، وهو نظام يغلف النظام المطمور عادة، ويضم إلى جانبه مكونات محسابية وميكانيكية أخرى. ويختلف النظام المطمور عن المحساب عمومي الأغراض (general purpose) في أن النظام المطمور لا ينجز مهاما متعددة كثيرة وإنما يقوم بأداء القليل من المهام المعينة المعرفة سلفا. والنظام المطمور ليس نظاما تفاعليا يعتمد على التشغيل البشري وإنما على وجود محساسات (sensors) ومشغلات (actuators). والأغلب ألا يحوي النظام المطمور وحدات تلاحم طرفية مثل لوحة المفاتيح والمرقاب (monitor) ما لم تكن هذه الوحدات ضرورية لتشغيل النظام الطامر.

يمكن تعريف النظام المطمور أيضا بأنه أداة هندسية تجري حسابات خاضعة لقيود فيزيائية. تنشأ هذه القيود من نوعين من تفاعلات العملية الحسابية مع العالم الفيزيائي هما: (أ) الاستجابة أو رد الفعل (reaction) للبيئة الفيزيائية، و(ب) تنفيذ الحسابات على منصة (platform) فيزيائية. ومن أمثلة قيود الاستجابة المواعيد النهائية (deadlines) وحجم الإنتاجية أو العطاء (throughput)، أما قيود الاستجابة فمنها السرعة والقدرة والمعولية.

تدخل النظم المطمورة في تكوين كثير من المنتجات المتقدمة، ويشمل ذلك مجالات عديدة مثل:

1. النقل (الطيران والفضاء والسيارات والقطارات)،
2. الأجهزة الكهربائية والإلكترونية (آلات التصوير ولعب الأطفال وأجهزة التلفاز والهواتف الخلوية)،
3. التحكم في العمليات (إنتاج وتوزيع الطاقة، ميكنة (أَتْمَتَة) وأَمْثَلة المصانع)،
4. الاتصالات (الأقمار الصناعية والشبكات الأرضية)،
5. الحماية والأمان (التجارة الإلكترونية والبطاقات الذكية).

إن الاستعمال المكثف والمتزايد للنظم المطمورة في منتجات الحياة اليومية يقود إلى تطور هائل في علوم وتقانة المعلومات، وليس بمستبعد أن تدخل هذه النظم في جميع المنتجات تقريبا خلال فترة وجيزة من الزمن. وفي البدء، كانت غاية مصممي النظم المطمورة إدراك الإمكانية (feasibility)، ولكنهم تعدوا ذلك الآن إلى تحقيق الأمثلة (optimization) باستهداف توصيل منتجاتهم بأقل تكلفة وفي أقصر زمن ممكن إلى القطاعات المعنية من السوق. تعني الأمثلة التكامل مع البيئة الفيزيائية والإلكترونية في احترام لما تفرضه الحياة العملية من قيود مثل قيود الموعد النهائي (deadline) والمعولية (reliability) والمتاحية (availability) والرسوخ (robustness)، واستهلاك الطاقة والتكلفة. والتوجه السائد الآن هو تحويل بنية النظم المطمورة من البنى الممركزة (centralized) إلى البنى

الموزعة (distributed) بما يحققه ذلك من تصميم منظم يرتكز على الفقرية (الوحدية) (modularity)، والتوزيع الأفضل لقدرة المعالجة، و إطاقة الخلل (fault tolerance).

تشتمل هذه الورقة على ستة فصول أولها فصل المقدمة الحالي. يسرد الفصل الثاني الخصائص المميزة للنظم المطمورة. ويصف الفصل الثالث عملية تصميم النظم المطمورة، فيبدأ بمقارنة تصميم الماديات والبرمجيات، ثم ينتقل إلى مفهوم التصميم المشترك لهما، كما يتعرض للفجوة القائمة بين هندسة النظم الحرجة وهندسة الجهد الأفضل. ويتبع ذلك الفصل الرابع متناولا التقدم المستمر في تقانة الدوائر المكاملة لأشباه الموصلات واستهداف هذا التقدم لوضع نظام على رقاقة. ويجري هنا الحديث عن انتقال النموذج الإرشادي من أسلوب الدوائر المكاملة محددة التطبيق إلى أسلوب التصميم المبني على منصة. يسعى الفصل الخامس لاستكشاف أهم المجروحيات التي تعاني منها النظم المطمورة وبيان كيف تتم الدراسة الكمية لهذه المجروحيات باستخدام شجرة الأخطاء، وكذلك توضيح كيفية حساب وتحسين معولية النظم المطمورة. وأخيرا يقدم الفصل السادس خاتمة لورقة البحث.

2. خصائص الأنظمة المطمورة

تتسم النظم المطمورة بأنها [1-5]:

1. أنظمة محسابية كاملة، يتداخل ويتكامل فيها العتاد المادي (الصلب) (hardware) مع العتاد البرمجي (الرخو) (software) المناظر في بيئة واحدة، وليس ميسورا التمايز والانفصال بين هذين العتادين.

2. أنظمة مكرسة (dedicated) لمهمة أو مهام قليلة تناط بها وتوقف عليها وتتخصص فيها، وهي في ذلك تختلف عن النظم عمومية الأغراض (general purpose) التي تتسم بالمرونة وتتكفل بمدى واسع من احتياجات المستخدم وتقوم بوظائف ومهام كثيرة ومتعددة.

3. لا تصمد بذاتها كأنظمة منفردة (standalone)، فكثير منها يوضع داخل نظام أشمل وأكمل وأكبر له أغراض أكثر عمومية (وهذا هو السبب في تسميتها بالنظم المطمورة أو المُبْطَنَة أو المُحتواة أو المدفونة). ومن أمثلة ذلك الأنظمة المطمورة المستعملة في ضبط سرعة المركبات والسيارات، والأنظمة المطمورة المستخدمة في موالفة أوتار بعض الآلات الوترية.

4. غالبا ما يقيد عملها بكونه في الزمن الحقيقي (real time)، بمعنى أن تكون أزمنة استجابتها لطلبات الخدمة أقل من قيم صغيرة معرفة سلفا، وذلك لأغراض تتعلق بالأمان ويسر الاستعمال.

5. تعليمات البرامج التي تشغلها تنتمي إلى ما يسمى العتاد الوسيط أو المتماسك (middleware or firmware) الذي هو وسط بين العتادين المادي والبرمجي، لكونه يخزن في الذاكرات القابلة للقراءة فقط (read-only memory) أو في رقاقات ذاكرات الوميض (flash-memory chips). ويتم تشغيل هذه التعليمات بالحد الأدنى من موارد العتاد المادي مثل الذاكرة ووحدات الدخل والخرج.

6. تستعمل كوسائل تحكم، ولكنها بدورها مُتَحَكَّم فيها من قِبل نبائط أخرى مثل المحكام الدقيق (microcontroller) ومعالج أو معلاج الإشارات الرقمية (digital signal processor).

7. تكريسها لأداء مهام محددة قليلة يجعل من الميسور على مصمميها توفير الأَمْثَلَة (optimization) بتصغير الحجم والكلفة وتعظيم المعولية والأداء. كما يمكن تخفيض الكلفة الإنتاجية للوحدة عند إنتاجها بالجملة.

3. تصميم النظم المطمورة

تصميم النظام هو عملية تشتق من متطلبات النظام نموذجا (تمثيلا مجردا مفيدا) له [6]، يمكن من خلاله توليد أو إنشاء النظام بصورة تلقائية تقريبا. ويختلف تصميم النظم المطمورة عن النظم المحسابية التقليدية في أنه لا يقبل الانفصال بين البرمجيات والماديات، ومن ثم فهو نوع من التصميم المشترك (co-design). نناقش فيما يلي الاختلافات بين تصميم الماديات والبرمجيات ثم نتحدث عن التصميم المشترك الآني لهما.

3.1. تصميم الماديات والبرمجيات

تصمم نظم الماديات من مركبات مترابطة يغلب عليها التوازي [3, 7]، وتمثلها نماذج تحليلية (معادلات) مهمتها تحديد دوال الانتقال (transfer functions) لها، ووصف كيفية سريان البيانات عبر مركباتها العديدة. أما نظم البرمجيات فتصمم بدلالة مكونات تتابعية مثل الكينونات (objects) والعروق (threads) تتغير بنيتها حراكيا (بالإنشاء والحذف والهجرة)، وتمثل هذه النظم بنماذج محسابية (برامج) يتمدد معناها تشغيليا بواسطة آلة تنفيذ تجريدية قد تسمى بالآلة الافتراضية (virtual machine) أو بالآلة الذاتية (automaton). إن تصميم الماديات مبني على المعادلات (equation based) أما تصميم البرمجيات فيبنى على الآلة (machine based) [3].

إن نوعي التجريد اللذين تقوم بهما النماذج التحليلية والنماذج المحسابية يعتبران متعامدين (orthogonal) أي أنهما يختلفان اختلافا جذريا. فالنماذج التحليلية تصلح بداهة لمعالجة الآنية (concurrency) ومع القيود الكمية، ولكنها تجد صعوبة في التعامل مع انعدام الحتمية (non-determination) الذي ينشأ عن المواصفات الجزئية التي تتوفر تدريجيا، كما يتعذر عليها متابعة التعقيدات الحسابية. أما النماذج المحسابية فطبيعتها مساندة تسلسلات التجريد غير الحتمي، ولها قدرة فعالة على تناول التعقيدات الحسابية، وفي المقابل فهي ليست ميسرة للمعالجة الآنية أو لإعمال القيود الفيزيائية [3].

وتستهدف النماذج التحليلية والمحسابية نوعين مختلفين من متطلبات النظام. فالنماذج المحسابية للبرمجيات معنية بالمتطلبات الوظيفية كالخدمات المتوقعة وصحة التشغيل والملامح العامة المستقلة عن التنفيذ، أما النماذج التحليلية

للماديات فهي معنية باستمرار الأداء ورسوخه، ولذا تتعلق بقيم الموارد الفيزيائية مثل التردد واستهلاك الطاقة والتخلف الزمني ومتوسط الزمن حتى الفشل والكلفة.

إن الاختلافات سالفة الذكر بين الماديات والبرمجيات تؤدي إلى اختلافات جوهرية بين عمليتي التصميم في كل منهما. فتصميم الماديات يستخدم أساليب تحليلية قوية يمكن أتممتها، أما تصميم البرمجيات فيسمح بالتنفيذ المباشر ويوفر مجالا أكبر للموالفة والتنوع والأمثلة ولا زال يجري على نطاق فردي غير قابل للأتمتة.

يوضح جدول **1** ملخصا لأهم الفروق بين الماديات والبرمجيات وهي فروق تم تجاهلها تاريخيا عند بدء تصميم النظم المطمورة، حيث كانت معظم طرائق هذا التصميم تنقسم في أصلها إلى نوعين هما:

- الطرائق المبنية على اللغات (language-based methods)، وهي طرائق موروثة عن مجال تصميم البرمجيات، وتعتمد على بعض لغات البرمجة مثل لغة أدا (Ada) ولغة جافا-ر-ت (Java-R-T).
- الطرائق المبنية على التركيب (synthesis-based methods)، وهي طرائق موروثة عن مجال تصميم الماديات، وتعتمد على استخدام لغات برمجة خاصة تطوع البرمجيات لخدمة الماديات، ولذا تسمى اللغات الواصفة للماديات (Hardware Description Languages).

إن التوجه الحالي في تصميم النظم المطمورة، هو ما يعرف بالتصميم المشترك [7-9]، ويمكن وصف طرائقه إجمالا بالطرائق المبنية على النموذج (model-based methods)، وهذا هو موضوع الفصل التالي.

جدول 1: مقارنة بين نظم الماديات ونظم البرمجيات.

وجه المقارنة	نظم الماديات Hardware Systems	نظم البرمجيات Software Systems
العلاقة بين مركباتها التمثيل	تشابك يغلب عليه التوازي نماذج تحليلية (معادلات)	التتابع والتغير الحراكي نماذج محسابية (برامج)
العملية الأهم	إنشاء دوال الانتقال	إنشاء الآلية الذاتية
وصف تركيبها	سريان البيانات عبر المكونات	سريان التحكم عبر المكونات
مجال براعتها	المعالجة الآنية وإعمال القيود الفيزيائية الكمية	التجريد غير الحتمي والمواصفات الجزئية التدريجية والتعقيدات الحسابية
الاستهداف	استمرار الأداء ورسوخه	المتطلبات الوظيفية

3.2. التصميم المشترك:

التصميم المشترك (co-design) للماديات والبرمجيات هو مكاملة التصميم للنظم المنفذة بكل من عناصر العتاد المادي التصميم (الصلب) وعناصر العتاد البرمجي، حيث تكون عناصر النوعين في تفاعل دائم لإنجاز مهمة محددة. تزايدت الحاجة إلى استخدام التصميم المشترك بسبب تزايد التنوع (diversity) والتعقيد (complexity) في التطبيقات التي تستعمل النظم المطمورة ونظم الزمن الحقيقي ونظم الاستجابة المفاعلية (reactive)، فضلا عن النظم المخَلَّطة التي تجمع بين مكونات تناظرية وأخرى رقمية أو نظم الزمن المتصل ونظم الزمن المتقطع.

يختلف التصميم المشترك عن طرائق التصميم التقليدي في أنه يربط باستمرار بين دورة تطوير الماديات ودورة تطوير البرمجيات، حيث تؤثر القرارات المتخذة عند تصميم العتاد المادي تأثيرات هامة على أنشطة تصميم العتاد البرمجي، والعكس بالعكس. وهذا يعني أن مشكلة التصميم تعامل بأكملها كوحدة واحدة. إن اسم التصميم المشترك قد يفيد كونه تصميما متزامنا أو آنيا (concurrent) يجمع الماديات والبرمجيات في صعيد وإطار واحد، ولكنه يعني أيضا تنسيقا جيدا بين الجهود المشتركة لمصممين ينتمون إلى مجالات عدة.

ويمكننا تصور الفارق الرئيسي بين التصميم المشترك والتصميم التقليدي على النحو التالي: في التصميم التقليدي يجري التصميم من خلال الفصل المتعمد لتصميم الماديات وتصميم البرمجيات في مرحلة مبكرة جدا من العمل. وبسبب هذا الفصل، يتم تصميم الماديات والبرمجيات بأساليب مختلفة وأدوات متباينة وبواسطة طائفتين متمايزتين من المصممين. أما التصميم المشترك فينجز أكبر قدر ممكن من العمل قبل التفكير في وسيلة التنفيذ وتحديد كونها مادية أو برمجية. ويتأتى ذلك من خلال صياغة نموذج مشترك (common model) لعناصر الماديات والبرمجيات. ويتسم هذا النموذج بالتجريد (abstractness) والعمومية (generality) بحيث يسمح للمصمم بعمل معظم التصميم دون أم يحدد أو يلتزم بكيفية التنفيذ، ويمكن في المرحلة النهائية من التصميم تحديد هوية كل عنصر من العناصر المصمَّمَة بجعله عنصرا ماديا أو برمجيا أو خليطا منهما. فإذا كان العنصر متغيرا تم تنفيذه برمجيا حتى يمكن تعديله بسهولة عن طريق البرمجة، أما إذا كان تغير العنصر أمرا غير وارد مطلقا أو كان زمن أدائه يؤثر تأثيرا حرجا على النظام ككل، فإن من الأجدى والأنفع تنفيذه في الصورة المادية (الصلبة).

تعتمد طرائق التصميم المشترك (الطرائق المبنية على النموذج) للنظم المطمورة على إضافات في لغات البرمجة الإجرائية المعروفة، وعلى بعض اللغات الحديثة مثل اللغات المتزامنة (synchronous) [3] التي تشمل بنيتها البرمجية بعض المفاهيم المجردة للعتاد المادي، ومثل اللغات المتخصصة في النمذجة، وأشهرها هي لغة النمذجة الموحدة (Unified Modelling language) التي تعرف بالمختصر (ل ن و) (UML) [10-11] وكذا لغة تحليل وتصميم البنى (Architecture Analysis and Design Language) التي تسمى اختصارا (ل ح ص ب) (AADL) [12]. ولغات النمذجة هذه توفر استقلالية أكبر عن لغات البرمجة المعتادة وتركز على بنية النظام كوسيلة

لتنظيم الحسابات والاتصالات والقيود. وقد حققت هذه اللغات نجاحات لا بأس بها في ردم الهوة بين النماذج التحليلية للماديات والنماذج المحسابية للبرمجيات [3]. ولكن لا زالت هناك إشكالية أخرى نتعرض لها في الفصل التالي.

3.3. هندسة النظم الحرجة وهندسة الجهد الأمثل

تتراوح الممارسات الهندسية في عالم اليوم بين نقيضين هما:

(أ) هندسة النظم الحرجة (Critical-systems engineering) التي تحاول ضمان الأمان بأي ثمن حتى عندما يعمل النظام تحت ظروف شديدة الوطأة بالغة التطرف، وهذه الهندسة تنظر للتصميم على أنه مسألة استيفاء شروط والتزام بقيود محددة لا هوادة فيها. تعتمد هذه الهندسة على تحليل أسوأ الأحوال (worst-case)، ولذلك فجميع تقريباتها محافظة وحجزها للموارد ثابت لا يتغير. وليس مسموحا للمتاحية (availability) أن تقل عن مائة بالمائة، وبعبارة أخرى ليس مقبولا للنظام أن يفشل في أداء المهمة المنوطة به في أي وقت من الأوقات. وذلك لا يتأتى إلا من خلال استعمال وفرة كثيفة (massive redundancy)، أي استعمال عناصر كثيرة زائدة على الاحتياجات الوظيفية، وهذا يجعل النظام مطيقا للخلل (fault-tolerant) فيصير بمقدوره اكتشاف ما يقع فيه من أعطاب واستعادة التشغيل رغم وجود هذه الأعطاب.

(ب) هندسة الجهد الأمثل (Best-effort engineering): التي تسعى لبلوغ الوضع الأمثل لأداء وتكلفة النظام حينما يعمل تحت الظروف المتوقعة، وهذه الهندسة تنظر للتصميم على أنه مسألة أَمْثَلَة (optimization)، وهي تعتمد على تحليل الحالة المتوسطة (average-case) وعلى التحصيص الحراكي للموارد (dynamic resource allocation) بهدف رفع فاعلية وكفاية استخدام هذه الموارد. تصلح هذه الهندسة للتطبيقات التي تقبل بشيء من الاضمحلال (degradation) وتتحمل رفضا مؤقتا لتوفير الخدمة، وذلك يتحقق بواسطة آليات تضبط وتوالف معالم النظام حال تشغيله (اعتمادا على ملاحظة حالته من خلال إشارات تغذية مرتدة).

ويوضح جدول **2** تلخيصا لأهم أوجه الاختلاف بين هندسة النظم الحرجة وهندسة الجهد الأمثل. وهذه الاختلافات تعني وجود فجوة متسعة ومتزايدة بين هذين النوعين من الهندسة. ويعتقد الكثيرون أن هذه الفجوة لا مناص عنها ولا مهرب منها في كثير من الحالات، خاصة في حالة النظم المطمورة التي يراد تصميمها للعمل في بيئات يكتنف معلوماتها الكثير من النقص والريبة والغموض. وقد بدأت هذه الفجوة تعبر عن نفسها في صورة فصل فيزيائي بين الأجزاء الحرجة وغير الحرجة في كثير من النظم، حيث يتم تكريس عتاد مادي خاص أو فترات تشغيل زمنية محددة للأجزاء الحرجة وللأجزاء غير الحرجة كل على حدة. والمرجو أن ينجح التطور في علوم التصميم في ردم الفجوة سالفة الذكر أو تقليل اتساعها [3].

جدول 2: مقارنة بين هندسة النظم الحرجة وهندسة الجهد الأمثل.

وجه المقارنة	هندسة النظم الحرجة Critical Systems Engineering	هندسة الجهد الأمثل Best Effort Engineering
الهدف	ضمان الأمان بأي ثمن	أَمْثَلَة أداء النظام وتكلفته
طبيعة القيود	جامدة لا هوادة فيها (hard)	رخوة متسامح فيها (soft)
مشكلة التصميم	مسألة استيفاء شروط محددة	مسألة أَمْثَلَة
ظروف التشغيل	أسوأ الظروف	الظروف العادية (المتوسطة) المتوقعة
متاحية الخدمة	يتعين توفير أعلى قدرة حسابية بلا انقطاع في جميع الأوقات	مسموح بانقطاع الخدمة مؤقتا لفترات وجيزة لا تخل بحد أدنى من مستوى جودة الخدمة
أساس العمل	الوفرة الكثيفة والعناصر المطيقة للخلل	آليات الضبط والموالفة
أشهر أمثلتها	نظم الطيران	نظم الاتصالات

4. تصميم نظام على رقاقة (ن ع ر)

4.1. تصغير الدوائر المكاملة

اتسمت تقانة الدوائر المكاملة لأشباه الموصلات منذ نشأتها عام 1958 بالتصغير المتواصل في الأبعاد التي تستخدمها. وقد أخذ هذا التصغير طابعا أسيا وفقا لما لاحظه جوردون مور من أن عدد الترانزستورات التي يمكن وضعها بصورة اقتصادية على دائرة مكاملة يتضاعف كل عامين. وقد تنبأ مور أن تصدق ملاحظته لعقد واحد من الزمان على الأقل، ولكنها تجاوزت ذلك بكثير فظلت صحيحة حتى الآن، ولا يتوقع لها أن تخلف وعدها بحال قبل عام 2015. وقد أدى هذا النجاح لنبوءة مور إلى تسميتها بقانون مور (Moore's law). ويسري هذا القانون بصور مختلفة على العديد من المؤشرات والمقاييس الخاصة بالدوائر المكاملة، ويستخدم في التخطيط بعيد المدى في الصناعة وفي وضع أهداف البحث والتطوير. وبالطبع لن يصح قانون مور إلى ما لا نهاية لأنه سوف يصطدم في وقت ما ببعض الحدود الفيزيائية الأساسية مثل حجم الذرات وقانون الريبة (عدم التأكد) لهايزنبرج.

إن استمرار التصغير في تقانة أشباه الموصلات قد أسفر عن تحقيق فكرة النظام على الرقاقة (ن ع ر) (SoC-System on Chip). وهذه الفكرة تعني أن تضم رقاقة واحدة نظاما إليكترونيا كاملا بما في ذلك جميع وحداته الطرفية ونقاط اتصاله وتلاحمه مع العالم الخارجي. وبذلك سوف تصبح الإلكترونيات موجودة في كل مكان (ubiquitous) وتنشأ لها استعمالات كانت تعد من قبيل الخيال العلمي منذ زمن ليس بالطويل. مثال ذلك التطبيق

المعروف باسم الذكاء المحيط أو المكتنف (Ambient Intelligence)، وهو تطبيق يرتكز على شبكات واسعة الانتشار تتألف من نقاط للإحساس والتحكم والتشغيل مطمورة في بيئة الحياة اليومية بهدف تحسين عدد من الأنشطة الحياتية المألوفة والبيئة التي تضمها.

إن الحل المبني على الشبكة المسمى بحل الشبكة على رقاقة (ش ع ر) (NoC – Network on Chip) يبدو هو البديل الوحيد القادر على التعامل مع مشاكل التوازي الكثيف (massive parallelism)، بما فيها مشاكل المعولية والرسوخ والتزامن وإدارة القدرة. سوف يتحقق هذا الحل من خلال نقلة في النموذج الإرشادي من أسلوب الدوائر المكاملة المحددة التطبيق (د ك ح ط) (ASIC – Application Specific Integrated Circuits) إلى أسلوب التصميم على منصة (ص ع ن) (Design on a platform).

4.2. اضمحلال وسقوط الدوائر المكاملة محددة التطبيق

كانت منهجية تصميم الدوائر المكاملة محددة التطبيق ناجحة للغاية في استخدام الإلكترونيات الرخيصة في مدى واسع من التطبيقات، ولكنها بدأت في الاضمحلال خلال العقدين الماضيين للأسباب التالية [13]:

- التزايد في كلفة تصنيع الدوائر المكاملة مع دخولها منطقة ما تحت الميكرون (المنطقة التي تقل الأبعاد النمطية فيها عن ميكرون واحد أي عن واحد على مليون من المتر).
- تأثيرات المنطقة العميقة تحت الميكرون: مثل تأخير التوصيل البيني (Interconnect delay)، والتداخل، وضجيج المنبع، فضلا عن تبديد الطاقة والتسرب والمتغيرية (Variability)، وهذه التأثيرات جعلت عملية التحقق والاستيثاق (Verification) تستولي على أكبر نصيب من كلفة عملية التصميم.
- تعقدت عملية التصميم إذ أصبح من المألوف أن تتناول ما ينيف على مائة مليون ترانزستور، وهذا التعقيد اقتضى ضخامة حجم فريق المصممين أو زيادة كبيرة في الزمن المستغرق في عملية التصميم ذاتها.

4.3. التصميم على منصة

يعرف التصميم على منصة أيضا باسم التصميم المبني على منصة (Platform-based design)، كما يعرف في صورته المحسنة بالتصميم الممركز في منصة (Platform-centric-design). وتعرف المنصة بأنها عائلة من المنتجات تتسم بمجموعة من العناصر المشتركة (commonalties) توضع مواصفاتها ويتم تنفيذها بحيث يتسنى تهيئتها وتوجيهها للحصول على منتجات نهائية محددة. وتستعمل فكرة المنصات الآن في نطاق واسع من المنتجات الهندسية يشمل منتجات مختلفة كالطائرات والمحساب الشخصي والهاتف النقال [13-16]. ورغم أن الحرص على العناصر المشتركة للمنصة قد يقلل من الأداء ويعرقل الإبداع، فإن منهجية المنصة تظل مفيدة على الإجمال في عملية

تطوير المنتجات النهائية، لأن المنصة الواحدة توفر إعادة استخدام رتيب (systematic reuse) من شأنه توليد منتجات عديدة يمكن ترقيتها وتحسينها في وقت قصير وبكلفة يسيرة من خلال ترقية وتحسين المنصة ذاتها [13-16].

ورغم أن فكرة المنصات معروفة منذ عقود، فإنها لم تطرق باب صناعة الدوائر المكاملة إلا مؤخرا. ففي مجال الدوائر المكاملة أصبح من الممكن وصف المنصة بأنها دائرة مكاملة مرنة (flexible) يمكن تكييفها وتأهيلها لتطبيق معين من خلال برمجة واحد أو أكثر من عناصرها. وقد تطورت فكرة المنصة في الدوائر المكاملة كثيرا حتى وصلت إلى منهجية التصميم الممركز في منصة التي تتسم بتفادي إعادة تصميم الرقاقة وتصنيعها من الصفر، وذلك من خلال إعادة استخدام رقاقات مصممة سلفا (pre-designed). كذلك تتسم هذه المنهجية باستخدام أفضل وأحسن أدوات ولغات النمذجة المتاحة لعمل التصميم المشترك للعتاد البرمجي والمادي (الذي يغلب عليه شيء من التحيز للبرمجيات (software biased))، بهدف تطوير نظام على رقاقة بصورة صحيحة وسريعة.

إن منهجية التصميم على منصة تعتمد على طبقات متراكمة من التجريد (abstraction) تفصل عالم التصميم عن عالم التنفيذ، ومن ثم يمكنها أن تستوعب تقانات للتنفيذ شديدة التغاير والاختلاف. إن حل المنصة هو حل وسط بين الطرائق التي تهبط من القمة لأسفل (top-down) وتلك التي تصعد من القاع لأعلى (bottom-up). إن تصميم المنصة هو تنقيح متعدد الخطوات للمواصفات وصولا بها إلى تجريد أدنى بالاختيار من مجموعة محددة من المكونات المتاحة سلفا [13].

إن أسلوب المنصة لن يقتصر على جانب الحسابات بل سوف يمتد ليشمل جانب الاتصالات، وهذا ما يمثل الانتقال إلى مفهوم الشبكة على رقاقة. غير أن مهمة تصميم الشبكات على رقاقة لن تكون ميسورة بحال فقد تسفر عن تنفيذ غير سليم للبروتوكولات (protocols) المستخدمة يجعلها تخطئ في أداء وظيفتها أو تؤديها على نحو دون المطلوب كأن تمعن في استنفاد الطاقة أو تسبب تأخيرات زمنية غير محتملة. وسوف يتم تصميم نظم الاتصالات على رقاقة (communication on chip) من خلال تعميم مفهوم الطبقات (layers) المستخدم بصورة واسعة ومقننة في نظم شبكات الاتصالات المعتادة. من المتوقع أن ترث نظم الاتصالات على رقاقة الطبقات السبع التي يقننها النموذج الإسنادي (reference model) للتوصيل بين النظم المفتوحة (open systems interconnection)، وهي بداية من أسفل إلى أعلى: الطبقة الفيزيائية (physical)، وطبقة وصلة البيانات (data link)، وطبقة الشبكة (network)، وطبقة الانتقال (transport)، وطبقة الانعقاد (session)، وطبقة العرض (presentation)، وطبقة التطبيق (application).

5. المجروحية والمعولية

5.1. مجروحية النظم المطمورة

تتسم النظم المطمورة بالمجروحية (vulnerability) أي بقابليتها للجرح أو الانجراح أو العطب أو الاختراق. وتأخذ مجروحية النظام المطمور صورا عدة أهمها: (أ) استنفاد منبع الطاقة الذي يغذيه، وهو غالبا بطارية محدودة، ويتم ذلك بزيادة الحمل الحسابي أو اختزال دورات النوم (sleep cycles) أو زيادة استخدام الوحدات الطرفية كالمحساسات، (ب) التطفل والاقتحام والعبث، حيث يؤدي موقع النظام المطمور أحيانا إلى تيسير ولوجه من قبل بعض المعتدين، و (ج) سرقة المعلومات والإخلال بالخصوصية، في حالة توصيل النظام بشبكة الشبكات، و (د) اختطاف النظام، أي استخدامه في غير ما قصد له [5]. ويضاف إلى صور المجروحية سالفة الذكر الصور التقليدية لفشل المكونات البرمجية والمادية، وهي صور متفاقمة نتيجة لقيود التشغيل غير المألوفة في النظم الأخرى.

5.2. استعمال شجرة الأخطاء

يستعمل أسلوب شجرة الأخطاء في هندسة المعولية بغرض التكمية (التقدير الكمي) لمجروحية النظم المطمورة [17]. ويمكن وصف شجرة الأخطاء بأنها مخطط منطقي مدخلاته تمثل أحداث الانجراح عند مستويات النظام المختلفة، ورؤوسه تمثل عمليات أو بوابات منطقية، أما خرجه (الذي هو جذر شجرة الأخطاء) فيمكن أن يكون أيا من الأحداث الأوجية غير المرغوب فيها. توجد خوارزميات عديدة لتحويل التعبير التبديلي (البولاني) لمتغير البيان للحدث الأوجي إلى تعبير احتمالي. وبالحصول على قيمة احتمال الحدث الأوجي، يمكن ضرب هذه القيمة في قيمة مجروحية النظام بهذا الحدث للحصول على قيمة كمية لانكشاف النظام أمامه. يرتبط استعمال شجرة الأخطاء أيضا بمعالجة المسألة مزدوجة العشوائية الخاصة بتقدير الريبة في احتمال الحدث الأوجي، وذلك باستخدام صيغة تحليلية دقيقة تربط التباين في احتمال الحدث الأوجي بالتباينات في احتمالات الأحداث الأساسية [18-19]. يفيد استعمال شجرة الأخطاء أيضا في عمل ترتيب لأحداث الانجراح المختلفة فيها حسب أهميتها.

5.3. دراسة المعولية

تختلف دراسة معولية النظم المطمورة عن دراسة معولية غيرها من النظم في أنها تضم مكونات برمجية ومادية يجب أخذ معولياتها في الحسبان، وفي أن معوليات هذه المكونات تعتمد إحصائيا على بعضها البعض. ومن التبسيط المخل الاقتصار على المكونات البرمجية (بافتراض معولية تامة (perfect) للمكونات المادية)، أو اعتبار معوليات المكونات

مستقلة إحصائيا عن بعضها البعض. ففي واقع الأمر تكون معوليات المكونات المادية غير تامة، ولا تستقل عن المكونات البرمجية، بل إنها تضمحل كلما زاد تحميلها بمكونات برمجية [20]. ومن التطورات الهامة أن تؤخذ المعولية في الاعتبار عند بدء التصميم وليس بعد الانتهاء منه، بحيث يستخدم التصميم المشترك للماديات والبرمجيات وفرة (redundancy) بصفة خاصة في المكونات الحرجة أو الهامة [9]. كذلك زاد الاهتمام عند تصميم النظم المطمورة بأن تكون مطيقة للخلل أو قادرة على اكتشاف الخلل ثم إصلاح نفسها بنفسها [21].

6. خاتمة

حاولنا في هذه الورقة أن نقدم للقارئ العربي استعراضا سريعا لما تحفل به أدبيات النظم المطمورة. قمنا بسرد الخصائص المميزة للنظم المطمورة مبرزين خصائص تكريسها وتداخل مادياتها وبرمجياتها وتقيد عملها بالزمن الحقيقي. أعطينا وصفا لعملية تصميم النظم المطمورة، حيث بينا أنها لا تندرج تحت تصميم الماديات وحدها ولا تصميم البرمجيات وحدها، ولكنها عملية تصميم مشترك آني لهما، كما تعرضنا للفجوة القائمة بين هندسة النظم الحرجة وهندسة الجهد الأفضل، وتأثير هذه الفجوة على عملية تصميم النظم المطمورة. استعرضنا تاريخ التقدم المستمر في تقانة الدوائر المكاملة لأشباه الموصلات واستهداف هذا التقدم لوضع نظام على رقاقة، كما تحدثنا عن انتقال النموذج الإرشادي من أسلوب تصميم الدوائر المكاملة محددة التطبيق إلى أسلوب التصميم المبني على منصة، وهو أسلوب دخل عالم الدوائر المكاملة مؤخرا بعد أن حقق نجاحات باهرة في ميادين هندسية عديدة أخرى. سردنا أهم المجروحيات التي تعاني منها النظم المطمورة، وبينا كيف تتم الدراسة الكمية لهذه المجروحيات باستخدام شجرة الأخطاء، وكذلك أوضحنا كيفية حساب وتحسين معولية النظم المطمورة.

وإذا كان الهدف الظاهر لورقة البحث هذه هو مراجعة واستعراض أدبيات النظم المطمورة، فإن الهدف الباطن لها هو استحداث مصطلحات عربية لما استجد في هذه الأدبيات، مع وضع مختصرات (abbreviations) مناسبة لهذه المصطلحات تعتمد على جذور الكلمات المستعملة. وكعمل مستقبلي فإننا نطمح إن شاء الله إلى كتابة بحث باللغة العربية يتضمن إسهاما جديدا في دراسة معولية النظم المطمورة، وذلك من خلال تطوير عملنا السابق في تحليل معولية النظم الوافرة جزئيا [22-23] ليغطي حالة الاعتماد الإحصائي بين المكونات. كذلك نود أن نستكشف التطبيقات المحتملة للهندسة العكسية في مجال النظم المطمورة، علما بأن الهندسة العكسية مطبقة على نطاق واسع في مجالي الماديات والبرمجيات على حد سواء [24].

المراجع

[1] **Bouyssounouse**, **B.**, and **J. Sifakis** (Editors), *Embedded Systems Design: The ARTIST Roadmap for Research and Development*, Springer-Verlag, Berlin-Heidelberg, Germany, 2005.

[2] **Mayer-Linderberg**, **F.**, *Dedicated Digital Processors: Methods in Hardware/Software System Design*, John Wiley & Sons, Ltd., Chichester, UK, 2005.

[3] **Henzinger**, **T. A.**, and **J. Sifakis**, The Embedded Systems Design Challenge, in **J. Misra**, **T. Nipkow**, and **E. Sekerinski** (Editors), *Proceedings of the 14th International Symposium on Formal Methods(FM 2006: Formal Methods*), pp. 1–15, Springer-Verlag, Berlin-Heidelberg, Germany, 2006.

[4] **Colnarič**, **M.**, **D. Verber**, and **W. A. Halang**, Distributed Embedded Control Systems: Improving Dependability with Coherent Design, Springer-Verlag, London, UK, 2008.

[5] **Parameswaran**, **S.**, and **T. Wolf**, Embedded Systems Security—an Overview, *Design Automation of Embedded Systems*, Vol. **12,** pp. 173–183, 2008.

[6] **Starfield**, **A. M.**, **K. A. Smith**, and **A. L. Bleloch**, *How to Model It*: *Problem Solving for the Computer Age*, Interaction Book Company, Edina, MN, USA, 1994.

[7] **Wirth**, **N.**, Hardware/Software Co-design Then and Now, *Information Processing Letters*, Vol. **88**, pp. 83–87, 2003.

[8] **Schrott**, **G.**, and **T. Tempelmeier**, Putting Hardware-Software Codesign into Practice, *Control Engineering Practice*, Vol. **6**, pp. 397–402, 1998.

[9] **Xie**, **Y.**, **L. Li**, **M. Kandemir**, **N. Vijaykrishnan** and **M. J. Irwin**, Reliability-aware Co-synthesis for Embedded Systems, *Journal of VLSI Signal Processing*, Vol. **49**, pp. 87–99, 2007.

[10] Martin, **G.**, and **W.** Muller (Editors), *UML for SOC Design*, Springer, Dordrecht, The Netherlands, 2005.

[11] **Duc**, **B. M.,** Springer, *Real-Time Object Uniform Design Methodology with UML*, Dordrecht, The Netherlands, 2007.

[12] **Feiler**, **P. H.**, **B. Lewis**, and **S. Vestal**, The SAE Architecture Analysis and Design Language (AADL) Standard: A Basis for Model-Based Architecture-Driven Embedded Systems Engineering, *Proceedings of the RTAS Workshop on Model-Driven Embedded Systems*, pp. 1–10, 2003.

[13] **Rabaey**, **J. M.**, System-on-Chip Challenges in the Deep-Sub-Micron Era - A case for the network-on-a-Chip, in **J. Nurmi**, **H. Tenhunen**, **J. Isoaho**, and **A. Jantsch** (Editors), *Interconnect-Centric Design for Advanced SoC and NoC*, pp. 3–24, Kluwer Academic Publishers, New York, NY, USA, 2005.

[14] **Madisetti**, **V. K.**, and **C. Arpikanondt**, *A Platform-Centric Approach to System-on-Chip (SOC) Design*, Springer, New York, NY, USA, 2005.

[15] **Bailey**, **B.**, **G. Martin**, and **T. Anderson** (Editors), *Taxonomies for the Development and Verification of Digital Systems*, Springer, New York, NY, USA, 2005.

[16] **Simpson**, **T. W.**, **Z. Siddique**, and **J. Jiao** (Editors), *Product Platform and Product Family Design - Methods and Applications*, Springer, New York, NY, USA, 2006.

[17] **Kaufmann**, L**. M.**, **J. B. Dugan**, **R. Manian**, and **K. K. Vemuri**, System Reliability Analysis of an Embedded Hardware/Software System using Fault Trees, *Proceedings of the Annual Reliability and Maintainability Symposium*, pp. 35-41, 1999.

[18] **Rushdi**, **A. M. A.** and **O. M. Ba-Rukab**, A Doubly-Stochastic Fault-Tree Assessment of the Probabilities of Security Breaches in Computer Systems, *Proceedings of the Second Saudi Science Conference, Part Four: Computer, Mathematics, and Statistics,* Jeddah, Saudi Arabia, pp. 1-17, 2004.

[19] **Rushdi**, **A. M. A.** and **O. M. Ba-Rukab**, Fault-Tree Modelling of Computer System Security, *International Journal of Computer Mathematics,* Vol. **82**, No. 7, pp. 805-819, 2005.

[20] **Wattanapongsakorn, N.**, and **S. P. Levitan**, Reliability Optimization Models for Embedded Systems with Multiple Applications, *IEEE Transactions on Reliability*, Vol. 53, No. 3, pp. 406-416, 2004.

[21] **Coyle**, **E. A.**, **L. P. Maguire, T. M. McGinnity**, Self-Repair of Embedded Systems, *Engineering Applications of Artificial Intelligence*, Vol. **17**, pp. 1–9, 2004.

[22] **Rushdi, A. M. A.**, Reliability of k-out-of-n Systems, Chapter 5 in **K. B. Misra** (Editor), *New Trends in System Reliability Evaluation,* vol. **16**, *Fundamental Studies in Engineering,* Elsevier Science Publishers, Amsterdam, The Netherlands, pp. 185-227, 1993.

[23] **Rushdi, A. M. A.**, Partially-Redundant Systems: Examples, Reliability, and Life Expectancy, *Electronic Proceedings of the 1st Conference of the Egyptian Engineering Association (EEA*), Riyadh, Saudi Arabia, 2009.

[24] **رشدي**، ع. م. ع.، نظرة تعليمية عامة إلى الهندسة العكسية، *السجل الإلكتروني لبحوث المؤتمر الأول لجمعية المهندسين المصريين*، الرياض، المملكة العربية السعودية، 2009.

Embedded Systems: Characteristics, Design and Reliability

Mahmoud Ali Muhammad Rushdi[1] and Ali Muhammad Ali Rushdi[2]

[1]Electronics and Electrical Communications Department,
Faculty of Engineering, Cairo University, Giza, Egypt,
mahmoud.rushdi@gmail.com
[2]Faculty of Engineering, King Abdulaziz University,
P. O. Box 80204, Jeddah 21589, Saudi Arabia,
arushdi@kau.edu.sa

ABSTRACT. This paper presents a partial review and exposition of the literature on embedded systems, and explains through this exposition many important concepts pertaining to the design of engineering systems in general. Characteristics of embedded systems are listed, with a stress on their dedication, the non-separability of their hardware and software, and their being constrained to respond in real time. The design process of embedded systems is described. The process cannot be a process of hardware design alone or of software design alone, but must be one of concurrent co-design. The gap existing between critical-systems engineering and best-effort engineering is explained and its effect on the design of embedded systems is pointed out. The history of progress of semiconductor integrated circuits technology is outlined, and the goal of this progress is identified as the placement of a full system on a single chip. There is a paradigm shift from the design of application-specific integrated circuits to platform-based design. This latter type of design was adopted for integrated circuits after it had achieved dramatic successes in many other engineering disciplines. The vulnerabilities of embedded systems are listed, and their quantitative study via fault trees is briefly explained. Methods of evaluating and upgrading the reliability of embedded systems are also pointed out. Finally, the paper proposes the generalization of the reliability algorithms of partially redundant systems by relaxing the assumption of statistically-independent components, thereby allowing these algorithms to model the reliability of embedded systems. Another proposal of future work asks for the exploration of potential applications of reverse engineering in embedded systems.

Key Words: Embedded Systems, Co-design, Hardware, Software, Middleware, Critical Systems, System on Chip, Vulnerability, Reliability.