Essay, minor Embedded Systems
HHS Delft, december 2011
Door Ezra Neuteboom (09093095)
en Mark Schrauwen (20031137)

SystemC-AMS en Simulink; onafscheidelijk, afhankelijk of onverenigbaar?

De electronica wereld is groot en complex. Er zijn ontzettend veel elektronische componenten en zo ook manieren om tot een elektronisch product te komen. Het is niet raar dat er in de loop der tijd methodes zijn ontwikkelt om "gemakkelijker" een weg te banen door deze complexe wereld. Er is veel vraag naar strategieën om sneller naar een oplossing of ontwerp te geraken. De meest gebruikte en misschien ook wel meest intuïtieve methode is de "evolutionaire ontwikkel methode" [1]. Maar net zoals in de natuur kost een dergelijke manier van ontwerpen veel tijd. Het nadenken over, het bewust zijn van de parameters én het maken van adequate modellen van een systeem of ontwerp versnellen een dergelijk proces aanzienlijk [bron 9].

ESL

Een opkomende elektronica ontwerp methodologie is ESL of *Electronic system level.* Helaas is ESL niet strak gedefinieerd. Kortweg komt het er op neer dat bij het gebruik van ESL-ontwerp het *gedrag* van een systeem wordt vastgelegd in een hogere programmeertaal zoals C++ of MATLAB. Daaruit volgt (als het goed is) een beter begrip over het ontwerp of systeem, wat de kansen op een succesvolle implementatie vergroot op een kost-effectieve manier terwijl wordt voldaan aan de noodzakelijke eisen (bron 10). Vervolgens wordt steeds dieper en specifieker in gegaan op het systeem. Een sterk punt van ESL als methodiek is dat het niet is gebonden aan één discipline. Een hardware ontwerper kan ESL net zo goed toepassen als een (embedded) software ontwerper. ESL staat dus concurrent (gelijktijdig) ontwerpen toe. Al tijdens het hardware ontwerp kan een software ontwikkelaar beginnen aan de embedded software.

Één van de programeertalen die vaak wordt geassocieerd met ESL is SystemC. Het is set van C++ klassen en macro's waarmee het mogelijk is om:

- Op een hoog niveau een systeem te ontwerpen
- Hardware te beschrijven
- Embedded software te schrijven
- Concurrent te ontwerpen
- Te simuleren

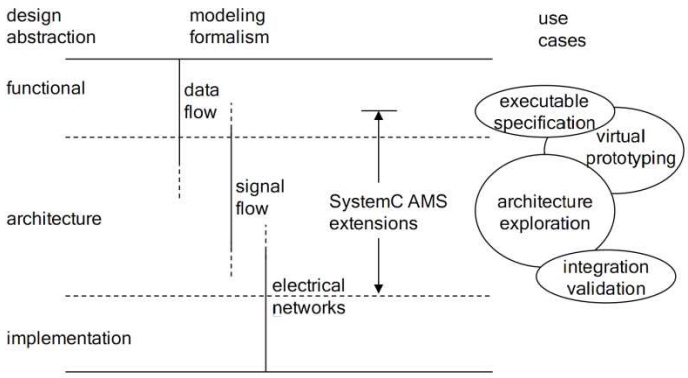
SystemC (SC) beperkt zich tot het digitale domein. Hoewel de elektronica wereld overheersend digitaal is, is de wereld van nature analoog. Goed en volledig system level design[2], met name op een hoger abstractie niveau, zou zich niet moeten beperken tot het digitale domein. Bij bijna elk elektronisch apparaat kan een elektronica ontwerper niet heen om bepaalde analoge aspecten. Voorbeelden van componenten of systemen die vaak niet mee gemoduleerd of gesimuleerd worden zijn sensoren en actuatoren, draadloze of bedrade fysieke lagen van communicatie blokken of de voeding van een geïntegreerd circuit [bron 14]. Het moge duidelijk zijn; om iets realistisch en volledig te kunnen modelleren en ontwerpen moet ook het analoge aspect van een model, ontwerp of systeem worden meegenomen.

[1] *Student projecten inbegrepen*
[2] *Het ontwerpen op systeem niveau*

Om de behoefte van de elektronica industrie te beantwoorden, is SystemC AMS (SCAMS) opgesteld. AMS staat voor Analog/Mixed Signals. Met behulp van SCAMS is het mogelijk om op een hoog abstractie niveau AMS functioneel gedrag te modelleren, architectuur te onderzoeken, de communicatie tussen digitaal en software te valideren en virtuele prototypes van elektronische systemen te maken. Of, in een andere bewoording, zoals staat geschreven in bron 3: *The basic idea behind SystemC AMS is right - mixed-signal design and verification need to move to higher levels of abstraction in order to run much faster. "When we talk about a system strategy, we need to make sure we include analog," said Andreas Kuehlmann, director of Cadence Research Labs. "To simulate any functionality, you need to simulate analog components together with software, processors, DSPs and so on."*

SCAMS werkt met drie verschillende rekenmodellen (Models of Computation, MoC) om het gedrag op verschillende abstractieniveaus te modelleren. Deze niveaus zijn:

- *Electrical linear networks (ELN)* zijn van essentiële waarde voor een systeem. Dit is nodig voor het modelleren van belastingen, beschermings- en voedingscircuit. De simulatie vindt plaats aan de hand van differentiaal- en algebraïsche vergelijkingen. Dit vindt plaats in het continue tijddomein.
- *Linear Signal Flow (LSF)* modellen beschrijven AMS systemen als controle systemen en filters op functioneel en architectuur niveau. Het is een continue (i.p.v. een discreet) model.
- *Timed data flow (TDF)* modelleert DSP algoritmes en communicatie systemen op functioneel en architectuur niveau. Dit gebeurt met discrete stappen in de tijd. [bron 1][bron 15]



Figuur 1: Het gebruik van SystemC-AMS met de use cases erbij.

In figuur 1 is te zien hoe SCAMS een link vormt tussen het digitale domein en het analoge domein. Van een hoog naar laag abstractie niveau krijg je: TDF, LSF en ELN.

SCAMS is een hulpmiddel door en voor ontwerpers [bron 20]. Het richt zich vooral op systeem ontwerpers die een systeem globaal willen modelleren, ontwerpers die aan systeem architectuur werken. Een systeemontwerper ontwerpt dan simuleerbare specificatie's van het systeem, een globaal ontwerp van het systeem, wat uiteindelijk verfijnd zou kunnen worden door een specialist. De doelgroep is dus vooral ontwerpers die werken met de focus op AMS/RF blokken. [bron 20][bron 21]

Waar SCAMS toegepast zou worden om zowel architectuur als functionele modellen te maken, wordt in de praktijk veelal Matlab van MathWorks gebruikt. Echter is Matlab niet zo toegespitst op zowel architectuur als functionele beschrijvingen. Matlab beperkt zich, voor het overgrote deel, tot architecturele beschrijvingen. Bijvoorbeeld, de opbouw van een analoge low pass filter in plaats van

het gedrag van een analoge low pass filter. Dit wil niet zeggen dat een functionele beschrijving met behulp van Matlab niet mogelijk is. Maar kijkend naar systeem ontwerp beperkt Matlab zich hoofdzakelijk tot de architectuur.

Een handige tool van Matlab is Simulink wat gebruikt wordt voor multi-domein simulaties; een model gebaseerd ontwerp voor dynamische en embedded systemen. Het biedt een interactieve, grafische omgeving en een aanpasbare set van block libraries waarmee je kunt simuleren, implementeren en testen van tijdafhankelijke systemen, waaronder communicatie-, control- en signaal bewerkingssystemen[bron 15]. Bij het ontwerpen van een systeem wordt Simulink gebruikt om het algoritme te modelleren om te controleren. Nadat het algoritme geverifieerd is, kan het systeem geïmplementeerd worden. Met de HDL-coder van Simulink is het bijvoorbeeld mogelijk om voor bepaalde blokken een synthetiseerbare HDL beschrijving te genereren.

Stelling

Uit een aantal onderzoeken is gebleken dat Mathworks Simulink (SL[3]) en SCAMS vaak gecombineerd wordt gebruikt. Er wordt dan gesproken over co-simulatie of codesign. Zo wordt SCAMS vaak gecombineerd met SL. Soms om sneller en nauwkeuriger te simuleren [bron 16], dit heeft meestal te maken met de Beta-status van SCAMS. Maar vaak ook omdat SL bepaalde voordelen heeft over SCAM. Zo zijn er tools ontwikkeld om SL-modellen om te zetten naar SC en SCAMS [bron 17]. In sommige onderzoeken is gekeken of SCAMS simulaties hetzelfde resultaat opleverde als SL simulaties [bron 18]. Daaruit blijkt dat de SCAMS en SL simulaties eigenlijk niet met elkaar vergeleken kunnen worden omdat SL niet de mogelijkheid heeft om IP-modellen te modelleren. De redenen voor cos-imulaties zijn vaak gebaseerd op dat continous-time modelling in SCAMS nog niet perfect werkt, vooral in vergelijking met SL [bron 19]. De meeste onderzoeken zijn het met elkaar eens dat SCAMS snellere simulatie toestaat dan SL. Tegelijkertijd wordt SL vaker als 'gouden standaard' gebruikt, wat begrijpelijk is gezien de Beta-status van SCAMS.

Daartoe komen we tot onze stelling:
"SystemC-AMS zal aansluiting moeten zoeken bij Matlab Simulink om succesvol te worden."

Deze stelling is tweeledig. Enerzijds kan hij zo worden opgevat dat wordt afgevraagd of SCAMS moet worden gecombineerd met SL en anderzijds kan hij worden opgevat dat wordt afgevraagd of SCAMS meer moet lijken op SL. In de eerste opvatting van de stelling wordt afgevraagd of er een wederzijdse afhankelijkheid bestaat tussen SCAMS en SL? Als dit zo is, waarom worden de twee dan niet samengevoegd? In de tweede opvatting van de stelling wordt afgevraagd of SCAMS net als SL zou moeten worden voorzien van een GUI. Op beide aspecten van de stelling zal worden ingegaan.

Om een mening te vormen over de stelling hebben wij onderzoek gedaan naar verschillende aspecten van SCAMS. Ook hebben we kort gekeken naar SL. We hebben verschillende artikelen gevonden waarin onderzoeken naar en met SC beschreven staan. De resultaten hiervan zijn besproken in de inleiding over SCAMS hierboven. Ook hebben we contact gezocht met verschillende onderzoekers en ontwikkelaars die betrokken zijn bij de standaardisatie van SC. Zo hebben we onder andere contact gehad met Alexander de Graaf, Senior Design Engineer aan de TU delft, met Martin Barnasconi, Product manager en cluster leider AMS/RF system design methods bij NXP Semiconductors en voorzitter van de SystemC-AMS Working Group, François Pecheux, Universiteit Pierre & Marie Curie, werkend in Laboratorium LIP6-SoC en Christoph Grimm, onderzoeker aan de TU Wenen en werkend aan design en design methodologie van embedded mixed-signal systemen.

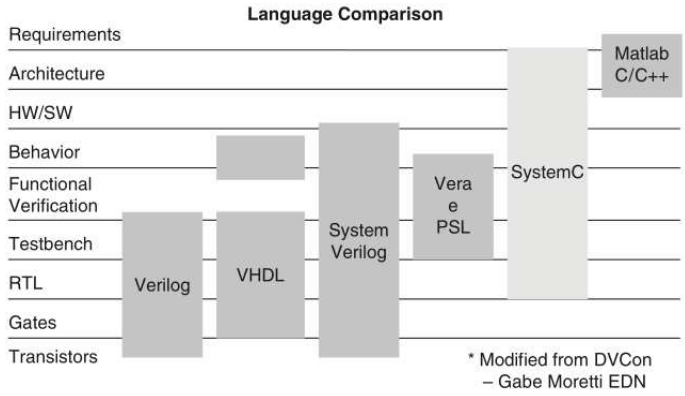
[3] *Met Simulink kan ook de programmeertaal Matlab worden bedoeld, dit wordt door elkaar gebruikt.*

Ook hebben we zelf getracht snel wat gebruikservaring op te doen met SCAMS en SL door een simpele filterschakeling te maken[4].

In de conversaties met de verschillende betrokkenen hebben wij verschillende onderwerpen en vragen met betrekking tot SCAMS besproken[bron 20][bron 21][bron 4].

Afhankelijkheid SCAMS en SL

Waaruit blijkt dat SCAMS überhaupt zou kunnen worden gecombineerd of met SL? Uit verschillende onderzoeken blijkt dat co-simulaties interessant zijn voor onderzoekers [bron 6][bron 17][bron 19][bron 22]. Ook uit het gesprek met Francois Pecheux is dit gebleken. Hij stelt dat dit met name interessant is om bepaalde onderdelen van een ontwerp nauwkeuriger te simuleren m.b.v. van SL. Een ander voordeel is dat als met veel verschillende tijd-domeinen[5] wordt gewerkt om te bekijken hoe een uitontwikkeld pakket als SL daar mee omgaat (interview). Ook blijkt dat continuous-time modelling niet direct mogelijk is [bron 19][bron 6]. Een bekende oplossing is dan co-simulatie. Oftewel, het gebruik maken van een software pakket die deze mogelijkheid wel ondersteunt. Een andere reden om SCAMS en SL gecombineerd te gebruiken is vanwege de behoefte aan een vergelijking. Het is duidelijk dat SCAMS en SL verschillend zijn. Maar ze vertonen ook overlap (zie figuur 2). Dan is het interessant om deze overlap te vergelijken [bron 22].



Figuur 2: het gebruik van programmeertalen in system level design (bron 12).

Het wordt duidelijk uit het voorgaande dat SCAMS en SL zijn in sommige opzichten afhankelijk van elkaar zijn van elkaar. SCAMS kan het feit dat SL betrouwbaar is uitontwikkeld, gebruiken als een onderdeel vergeleken kan worden met SL. Zo is het in SL net als in SCAMS mogelijk om stukjes architectuur of opbouw van een systeem te ontwerpen. Ook qua modellering in het continue tijddomein kan SL, SCAMS aanvullen. Andersom wordt duidelijk dat SL zeer beperkt is als vanuit system level design (inclusief de analoge aspecten) naar SL wordt gekeken. Martin Barnasconi bevestigd, ondanks zijn bevooroordeelde mening, dat SL niet goed kan omgaan met codesign: *"omdat deze tools meer in de functionele modeling zitten in niet goed overweg kunnen met AMS/HW/SW codesign vraagstukken die meer bij system en HW architectuur nodig zijn."* Dit wordt ook duidelijk uit figuur 2.

Met oog op de titel van dit document concluderen wij dat SL en SCAMS zeker *niet* onafscheidelijk en onverenigbaar zijn. Beide kunnen prima afzonderlijk worden gebruikt. Ze zijn tevens niet onverenigbaar omdat de verschillende co-simulaties en conversies het tegendeel bewijzen. Maar

[4] *Dit is gelukt in SL maar niet in SCAMS*

[5] *Vrij vertaald uit het interview: "The minute you have lots of different time constants, you are faced with how to deal with al these time scales. And using tools that have their own representation of time and own MoC will, their will be some losses in simulation performances is interesting*"

mag dan geconcludeerd worden dat ze wederzijds afhankelijk zijn? Wij denken van niet. Zoals al eerder in andere bewoordingen in dit document is vermeld staat SCAMS nog in kinderschoenen en loopt SL, bij wijze van spreken, rond in zeven-mijls-laarzen. Het zou vreemd zijn vanuit het oogpunt van SCAMS om geen gebruik te maken van SL, vooral nu het nog in ontwikkeling is. Maar hoe zit het dan met SL? Is SL dan afhankelijk van SCAMS? Wij denken van niet.

Mathworks is een bekend en groot bedrijf en SL wordt veel gebruikt in verschillende vakgebieden. SL kan prima voortbestaan zonder SCAMS. Maar voor een commercieel bedrijf als Mathworks zou de integratie van SCAMS alle voordelen van SCAMS opleveren (snelle simulaties, meer volledig systeem ontwerp). Waarom Mathworks dit nog niet heeft gedaan is ons een raadsel. Zo ook bij Martin Barnasconi: *"aangezien strikt genomen de "SystemC AMS extensions", de taal dus, door iedereen opgepakt kan worden als moderne taal voor het maken van AMS systeem beschrijvingen. Dus de heren van Mathworks zouden best de SystemC AMS taal kunnen gebruiken, maar blijkbaar hebben ze goede redenen om alleen maar hun eigen dialect (Matlab, systemobjects, ed) en dichtgetimmerde toolboxen/blocksets te leveren…"* Waarom Mathworks hier niet voor kiest heeft mogelijk te maken met de authenticiteit van Matlab en SL. Deze pakketten hebben altijd op zichzelf gestaan. Mogelijk ziet Mathworks het implementeren van SCAMS in SL als een ' vervuiling' van Matlab. Ook hebben wij vaker 'geproefd' bij de geraadpleegde experts dat er een soort van afkeer bestaat tegen Matlab. Kennelijk is Mathworks op te vatten als een soort closed sourced concullega van SCAMS. Een andere reden, waarom de experts onzes inziens afkeer ervaren, heeft mogelijk te maken met het ontwikkelen van een *standaard*. Vanwege vrije marktwerking principes kunnen overal SCAMS-achtige implementaties ontstaan. Een voorbeeld hiervan is de grote hoeveelheid en verscheidenheid aan programmeertalen die er zijn. In het ideale geval is er één doordachte en uitontwikkelde standaard die door iedereen wordt gebruikt. Als Mathworks bezig blijft met het implementeren van eigen standaarden wordt de ESL-wereld alles behalve overzichtelijker. Daarnaast draagt het closed sourced distribueren van dergelijke software niet bij aan innovatie.

SCAMS met een GUI

Is het noodzakelijk dat SCAMS net als SL een GUI krijgt? Als het aan de experts wordt gevraagd, lopen de meningen uiteen. Volgens Francois Percheux: "*I would say it would be interesting to show the people what contain you TDF cluster. Text coding is much more efficient.*[6]" Francois Percheux is volgens eigen zeggen: "*more of a text-guy"*. Volgens Alexander de Graaf [bron 23] is SCAMS juist gebaat bij een met SL vergelijkbare GUI. Vooral op een hoog abstractie niveau is het tekenen en verbinden van componenten een snelle en handige manier van ontwerpen. Maar uiteindelijk moet ook code in zulke componenten worden bewerkt en gezet. Een andere voordeel van een GUI is dat tijdens het ontwerpen van een GUI, onderdelen van de kennis, vaardigheden en ervaring van een SCAMS-gebruiker moeten worden gevangen in de GUI [bron 20]. Hetzelfde is natuurlijk gebeurd met hedendaagse tekstverwerkers. In het begin waren er mechanische typmachines, die vaak vastliepen en waarbij de mogelijkheid tot het corrigeren van tekst niet bestond. Uiteindelijk zijn we gekomen tot een programma als MS Word, waar ook dit document in is getypt, waar alle ervaringen, functionaliteiten en wensen van de gebruiker zo optimaal mogelijk in zijn verwerkt. De vraag is eigenlijk meer: "wat zijn precies de voordelen van een GUI"? Een GUI is vaak een alles-in-één-omgeving waarin de gebruikers alle tools tot zijn/haar beschikking heeft om hetgeen te doen waarvoor de GUI is ontworpen. Dit versnelt een proces aanzienlijk (zo ook bij het typen van documenten in MS Word). Ook maken GUI's code over het algemeen genomen inzichtelijker, vooral

[6] *Wij zijn ons bewust van het verkeerde Engels maar om het risico op een verkeerde interpretatie (en dus vertaling) te voorkomen, schrijven we letterlijk op wat Francois Percheux heeft gezegd.*

als dit wordt gecombineerd met visualisaties zoals flow-charts, state-machines, syntax highlighting, etc.

Wij denken dat SCAMS gebaat is bij een GUI. We willen deze stellingname illustreren met het volgende, stel jezelf de vraag: “Waarom is SCAMS überhaupt in leven geroepen”? Als het goed is kom je, in ieder geval na het lezen van het voorgaande, tot het antwoord dat in system level design (op een hoger abstractie niveau) de analoge ontwerp aspecten van het te ontwerpen systeem niet mogen worden genegeerd. Of nog abstracter; SCAMS, net als SC, is ontstaan uit de behoefte aan een system level design methodiek. Ontwerp methodieken als ESL zijn ontstaan omdat uit de praktijk blijkt dat zonder deze methodieken het bedenken, ontwerpen en produceren van elektronische apparatuur lang duurt en inefficiënt verloopt. Dergelijk methoden zijn er om de gehele productietijd in combinatie met de productie-efficiëntie te optimaliseren. SCAMS wordt, als alles goed gaat, in de toekomst een standaard die bij elk elektronisch ontwerp bedrijf kan worden gebruikt. Dit is de crux van onze stellingname; omdat SCAMS een hulpmiddel is, moet dit hulpmiddel in als zijn aspecten het ontwerp proces verbeteren en\of optimaliseren. Een belangrijk onderdeel is dan de wijze waarop dit hulpmiddel moet worden gebruikt. Een GUI is dus een eerste stap naar het toegankelijk maken van SCAMS.

Bron 1: Christoph Grimm et all (juni 2008). *An introduction to Modeling Embedded Analog/Mixed-Signal Systems using SystemC AMS Extensions.* Geraadpleegd op 15 december 2011, http://publik.tuwien.ac.at/files/PubDat_171466.pdf
Bron 2: Jean-François Boland et all. *Using MATLAB and Simulink in a SystemC verification environment*. Geraadpleegd op 15 december 2011.
Bron 3: Richard Goering. (18 maart 2010). *SystemC AMS – A New Proposal For Mixed-Signal Verification*. Geraadpleegd op 15 december 2011. http://www.cadence.com/Community/blogs/ii/archive/2010/03/18/systemc-ams-a-new-proposal-for-mixed-signal-verification.aspx
Bron 4: E-mail conversatie met Christopher Grimm op 28-11-2011
Bron 5: Anuja Apte. *Model and simulate event-driven systems.* Geraadpleegd op 15 december 2011. http://www.mathworks.nl/videos/simevents/esl-architectural-exploration-based-on-performance-analysis.html
Bron 6: F. Bouchhima et all. (2006). *A SystemC/Simulink Co-Simulation Framework for Continuous/Discrete-Events Simulation*. Geraadpleegd op 15 december 2011.
Bron 7: Yaseen Zaidi et all. (2009). *On Mixed Abstraction, Languages, and Simulation Approach to refinement with SystemC AMS.* Geraadpleegd op 15 december 2011.
Bron 8: Ralph Görgen et all. (2008) *Automatic Transformation of System Models in Automotive Electronics.* *http://www.offis.de/uploads/tx_useroffis/20091221111913_llncs.pdf* Geraadpleegd op 15 december 2011.
Bron 9: C. Verhoeven, et al. (2003) Structured Electronic Design.
Bron 10: Brian Bailey, Grant Martin and Andrew Piziali. (2007) *ESL Design and Verification: A Prescription for Electronic System Level Methodology*. Elsevier.
Bron 11: David C. Black et al. (2004) *SystemC: From the ground Up.* Kluwer.
Bron 12: AMS working group. Standard SystemC AMS Extensions 1.0. http://www.accellera.org/downloads/standards/systemc/ams/#main_reason
Bron 13:Christopher Grimm, et al. (June 2008). *An Introduction to Modeling Embedded Analog/Mixed-Signal Systems using SystemC AMS Extensions.* Geraadpleegd op 17 december 2011.
Bron 14: Martin Barnasconi. USA. *SystemC AMS Extensions: Solving the Need for Speed. Martin Barnasconi. USA.* Geraadpleegd op 17 december 2011.
Bron 15: Mathworks. Geraadpleegd op 17 december 2011 http://www.mathworks.nl/products/simulink/index.html
Bron 16: Kezhen Ma et al. (juni 2011) *A Fast and Accurate SystemC-AMS Model for PLL.* Geraadpleegd op 19 december 2011.
Bron 17*:* Dresden (mei 2011). *Automatic Transformation of MATLAB/Simulink Models to SystemC AMS*. http://www.accellera.org/news/events/systemc_ams_day/systemc_ams_day_2011_proceedings.pdf Geraadpleegd op 19 december 2011.
Bron 18: Michel Vasilevski et al. *Modeling Heterogeneous Systems Using SystemC-AMS Case Study: A Wireless Sensor Network Node.* http://www-soc.lip6.fr/~hassan/bmas07.pdf Geraadpleegd op 19 december 2011.
Bron 19: Philipp A. Hartmann et al. (11 oktober 2009) *Modelling control systems in systemc AMS- benefits and limitations.* Geraadpleegd op 19 december 2011.
Bron 20: Interview met Francois Pecheux. Gehouden op 12 december 2011
Bron 21: Interview met Martin Barnasconi. Gehouden op 8 december 2011
Bron 22: Ken Caluwaerts et al. (2008) *SystemC-AMS Heterogeneous modeling of a capacitive harvester of vibration energy*. http://www-soc.lip6.fr/~galayko/papers/bmas2008.pdf Geraadpleegd op 23 december 2011.
Bron 23: bezoek Alexander de Graaf, 19-12-11, TU-Delft

Bijlagen

Bijlage A: Interview met Francois Pecheux – Gehouden op 12 december 2011

Main question thesis: if SystemC-AMS needs to link up with Simulink. If SystemC-AMS needs a GUI like simulink.

1.What is the main purpose of SystemC AMS? We are interested in your view.

SCAMS missing link between lower level tools and higher level tools, SCAMS in conjunction SC operating, mixing, analog and foremost software. You can design and simulate highly original systems with integrated software. For me only software tool for building feedback tool with analog/digital components and software, that's my answer

2.What are the biggest advantages and disadvantages of SystemC AMS?

Biggest advantage is it's openes, you can try to modify the simulator. The main disadvantage the inner code is very complicated. The idea of building your own simulator is for me very interested. The question is not easy to answer. At the moment there is a lack of documententation which make it hard for new users

3.Who are the main target for SystemC AMS? Which kind of technicians or developers are going to use SystemC AMS? For example: hardware-developers, software-developers, etc.

For me, it's for system designers by system designers. This kind of people should make simulatable specification of a new system and then go to different specialist and present the simulatable specification and say this is what i want. The idea its a tool for engineer that design systems and engineers that do not want have accuracy and very precise models. They want the holisitc approach global solution that can be refined by specialist, the idea is to have all the disciplines within a single simulator. It's for people who want to do systems and not for specialist. It's a first order solution, that can interest the people from the financial side and the people from the inovative side.

4.What knowledge or skills does one need to have in order to be able to work with SystemC AMS?

Skills with plain C++. The most innovate model of SCAMS called TDF, I'm not interested in the other models. The idea is that with c++ and an idea of what an analog system is, technicians and engineers are quite, is possible fort hem to build AMS-systems, I would put C++ and some knowledge of analog and digital architequture, of course

5.What does SystemC AMS have what Mathwork's Simulink does not? Or to put it differently what are the main differences between SCAMS and SL?

I'm not specialist of SL. My answer is very biased. SL are commerical tools and cannot be used directly or simply. The idea is with any commerical tool its very important to absorb the knowledge of the designer and to integrate their work in their environment, with AMS it's the opposite.

6.Will SystemC AMS be needing a GUI in order to become succesfull?

I'm not convinced with GUI's. I'm more a text-guy. I think a schematic editor to build ptf clusters, could be of some help. Mainly to help a designer to find a good rate or delays to build feedback loops. I would say it would be interesting to show the people what contain you TDF cluster. Text coding is much more efficient.

7.What is new about SystemC AMS what e.g. VHDL AMS and Verilog AMS do not have?

It's not designed to do the same thing. For me AMS is to propose a solution for zero-order modelling. The idea is to put many disciplines together and to get an overview. It's not accurate because it's not designed to do what Verilog-AMS en VHD-AMS. These languages are much slower. The main point of SCAMS the model of computation (TDF) have been conveiced as quickly simulated with great efficientcy. It does not require building matrices modified model analysis techniques, it's much more simple. You put many accurate models in TDF but the main object is speeds and it's designed to simulate a complete RF, its not designed to very very accurated

8.Why would SystemC AMS succeed if other AMS languages do not?

Already answered

9.When looking for papers on SL and SCAMS a lot of the results that show up talk about co-designing and co-simulating with SL and SCAMS. That would suggests that some kind of co-operation is desired? What's your opinion?

I think it's interested way of simulating systems but it's much slower then a monolithic simultor. tHe idea when you make tools communciate you loose some efficienty in terms of simulation. Much more precise when you simulate at frequencies of 1 GHz. The minute you have lots of different time constants, you are faced with how to deal with al these time scales. And using tools that have their own representation of time and own MoC will, their will be some losses in simulation performances. Interesting way to refine models to applie to have one big block in SC and one tiny part refined in SL.

Bijlage B: E-mail conversatie met Christoph Grimm – Gehouden op 28 november 2011

From: Mark Schrauwen
question: SystemC-AMS Simulink

Hello Mr. Grimm,

We are students Electronics at the HHS Delft in the Netherlands. This semester we are following a minor called Embedded Systems. In this minor one of the things we have to do is to write an essay concerning embedded systems and we have the following thesis:

"SystemC-AMS needs to link up with Matlab Simulink to be successful"

On the site http://www.systemc-ams.org/educational.html is stated that you're an educator. It's highly likely that you possess a substantial amount of knowledge on the subject SystemC-AMS (especially in comparison to us). Would you be so kind to give your opinion (e.g. do you agree or disagree) on our thesis?

Because we have translated the thesis from Dutch to English the possibility exists that there is some kind of translation
error. So some specification is in order: we wonder if SystemC-AMS depends on Simulink for SystemC-AMS to be successful (e.g. to become an industry standard, much used, popular).

Thanks in advance.

Best regards,

Mark Schrauwen en Ezra Neuteboom
studenten Electronics
Minor Embedded Systems

Christoph Grimm wrote:
Dear Mark,

I think that SystemC AMS is already in some properties much *before* Simulink.
Simulink has undoubtedly the big asset of a nice, interactive UI, and good libraries.

However, SystemC AMS extensions allow real HW/SW Co-Design including analog/RF blocks.
Furthermore, it is better for those who like to “program”, not click.
The biggest asset of SystemC is that you can easily model digital HW and include IP blocks such as processors easily.

Just my 5 cent.
Nice evening,
Christoph

From: Christoph Grimm

Hi Mark,
SystemC AMS dosn’t depend on Matlab / Simulink at all.
It’s two competitive products …
Regards,
Christoph

From: Mark Schrauwen
Betreff: Re: question: SystemC-AMS Simulink

Hello Mr. Grimm,
I couldn't resist to reply on such a fast reply.
Your e-mail is clear. But I wonder, would you agree that SystemC-AMS is dependent on Matlab Simulink?
Do you think it's likely that SystemC-AMS will stand on its own? Or will electronic (HW/SW) designers always have the need for Simulink in combination with SystemC-AMS?

Bijlage C: E-mail conversatie met Martin Barnasconi – Gehouden op

Hi Mr. Barnasconi,

We are students Electronics at the HHS Delft in the Netherlands. This semester we are following a minor called Embedded Systems. In this minor one of the things we have to do is to write an essay concerning embedded systems and we have the following thesis statement:

"SystemC-AMS needs to link up with Matlab Simulink to be successful"? Or to put it differently: "SystemC-AMS needs a GUI (like Simulink) to be successful"?

On the site http://www.systemc-ams.org/educational.html is stated that you're the SystemC-AMS chairman. It's highly likely that you possess a substantial amount of knowledge on the subject SystemC-AMS (especially in comparison to us). Would you be so kind to give your opinion (e.g. do you agree or disagree) on our thesis statement?

An other highly interesting question about SystemC-AMS is: "For who is SystemC-AMS designed to be used"? Or to put it differently: "Which kind of technician is going to use SystemC-AMS"?

Your opinion on these questions would be extremely helpfull.

We hope to hear from you.

Best regards,

Mark Schrauwen and Ezra Neuteboom

Hallo Mark,

Ik neem aan dat het antwoord wel in het Nederlands kan.

Inderdaad, ik stuur de ontwikkeling van de SystemC AMS standardizatie. Let wel, de SystemC AMS extensions is een system-level beschrijvings taal gedefinieerd in een Language Reference Manual (LRM) en is dus geen “tool”. Fraunhofer heeft een implementatie, een simulator, onder de naam SystemC-AMS (met het streepje) en dit kan je zien als tool, maar is niet onderdeel van de OSCI AMS standard.

Om nu stellingen te deponeren gerelateerd aan een tool als Simulink is dus wat vreemd, aangezien strikt genomen de “SystemC AMS extensions”, de taal dus, door iedereen opgepakt kan worden als moderne taal voor het maken van AMS systeem beschrijvingen. Dus de heren van Mathworks zouden best de SystemC AMS taal kunnen gebruiken, maar blijkbaar hebben ze goede redenen om alleen maar hun eigen dialect (Matlab, systemobjects, ed) en dichtgetimmerde toolboxen/blocksets te leveren…

Het is dus aan de leveranciers van EDA tools om de taal te ondersteunen in een simulator, en daarnaast er ook een fatsoenlijke interface omheen te bouwen, vergelijkbaar wat met ander HDL talen ook het geval is (bijv SystemC, Verilog-AMS etc). Nu hebben groepen als Fraunhofer [1] en CISC [2] al dergelijke frameworks. De grote jongens (Mentor, Cadence, Synopsys) blijven een beetje achter wat dat betreft. Of ze snappen het niet, of ze zien geen business case (markt/klanten).

Ik zie zelf Matlab/Simulink niet als echt als bruikbaar platform voor integratie, omdat deze tools meer in de functionele modeling zitten in niet goed overweg kunnen met AMS/HW/SW codesign vraagstukken die meer bij system en HW architectuur nodig zijn. Overigens, Fraunhofer heeft ook hier oplossingen beschikbaar.

De stelling omtrent “wie is de doelgroep voor SystemC AMS” vind ik wel een interessante. Bij de toevoeging “AMS” denkt men al snel aan analoog (geen MS laat staan digitaal) design, en dus aan circuit designers. Dat is echter niet de doelgroep. Wanneer je wat verder leest hoe SystemC AMS werkt en wat je ermee kunt, dan kom je er achter dat de data flow model of computation (TDF) zeer interessant is voor abstracte beschrijving van bijv sampled analog/RF signalen. In dat opzicht lijkt het dus meer op SPW [3] en SystemVue [4]. Nadeel van deze tools is echter weer dat ze niet goed aansluiten bij de HW/SW en processor wereld, waar het gebruik van SystemC heel gangbaar is. De doelgroep is dus designers die meer aan system architecturen werken met de focus op AMS/RF blokken, architecten dus, die ook een gedegen kennis van SystemC hebben.

Hoop dat deze input helpt in het kiezen van de juiste stelling. Mochten er meer vragen zijn, dan hoor ik het wel.

Groeten,

Martin

[1] http://systemc-ams.eas.iis.fraunhofer.de/products/index_en.html
[2] http://www.cisc.at/system-architect-designer-2.html
[3] http://www.synopsys.com/systems/blockdesign/digitalsignalprocessing/pages/signal-processing.aspx
[4] www.agilent.com/find/eesof-systemvue