2012 International Conference on Image, Vision and Computing (ICIVC 2012)
IPCSIT vol. 50 (2012) © (2012) IACSIT Press, Singapore
DOI: 10.7763/IPCSIT.2012.V50.35

Abstraction Modeling of a Networked Senor System in a Multiple-Language Environment

Huajing Zheng [a +], Xiuli Sun [b], Xiaoyu Song [b]

[a] School of Optoelectronic Information University of Electronic Science and Technology, Chengdu, China
[b] Dept. of ECE Portland State University Portland, OR 97207, USA

Abstract. Developing highly efficient and reliable embedded system demands hardware/software co-design. To be high-configurable embedded systems, designers seek for transaction-level modeling and component based development. Our approach employ platform-based design and component-based design and apply it to a networked sensor system. We introduce a multiple-language environment unified component-based model and apply it to co-simulation of sensor system instances. The case studies demonstrate that our approach is readily applicable to real-world networked sensor systems and reduces the co-simulation complexities.

Keywords: high-Configurable; transaction-level; design; co-simulation

1. Introduction

In order to increase productivity, the design abstraction has been raised to the system level. Therefore, high level design decision can be made and reusability of components can be ensured. Transaction-level model (TLM) plays a key role in raising design abstraction. TLMs slice the de- sign process into small components. Each component carries its own functionality and can be validated by simulation. In TLM, the communication between the components is separated from the functional details of these components, which enables the reusability of components.

CBSD is a new type of software development. In contrast to traditional approach, CBSD is capable of building monolithic, single-platform, purpose-built-from-scratch systems to constructing assemblies of ready-made components which are platform-independent and supplied by third-parties. Different research is performed for CBSD such as theoretical foundations, through component models, and tool implementations.

Sensor systems have become increasingly important to our daily life. They can be found in most electronic devices, such as computers and automobiles. Sensor can sense and translate the sensed value into human readable data, such as temperature, sound, vibration, pressure, etc. A sensor network is a distributed network using sensors to monitor conditions in different environments. Such networks use a large number of small inexpensive self-contained devices. Each device is a sensor node equipped with a small computation unit, sensing unit, RF transceiver, energy source and embedded software.

In general, a sensor network system is an embedded sys-tem which contains both hardware and software(HW/SW). The interaction between HW/SW parts is crucial to the success of the system design. Traditional simulation-based verification is used to assure reliable systems. However, it is no longer sufficient. This is primarily due to simulation run- times lasting several hours and often days for a single design iteration as the size of simulations increases. On the other hand, HW has to be fabricated before testing with SW using trial and error method until the design meets the specification. In addition, the amount

[+] Corresponding author. Tel.: + (13981883136).
E-mail address: (279351026@qq.com).

of time consumed on the integration of hardware and software can take up more than half of the project cycle.

In this paper, we model the software and hardware for networked sensor systems in a multiple language environment with TLM. We perform co-simulation for the integrated system. Hence, we are capable to identify the design errors of the HW and SW at the early stages of the project.

The organization of this paper is as follows. Section 2 overviews the related work in this area. Section 3 defines our unified model development.Section 4 describes architecture of our sensor networked system by applying unified model development method. Conclusion is given in Section5.

2. Related Work

TLM first appears in system language and modeling domain.The TLMs are timed algorithmic descriptions. It describes hardware only and is generally used for bus modeling by connecting various devices. Since it does not contain pin level detail, the simulation is performed much faster than the lower level modeling such as Register Transfer Level (RTL).UCI [3] defines the construct of channel. It enables separation of communication and computational unit.[4] defines the feature in TLM with SystemC[11]supporting the channel.

CBSD [5]is an approach potentially reduce development costs and assemble systems rapidly by reusing previously built software components in a large system. The foundation of this approach assumes common parts in a system should be written once and assembled through reuse. In CBSD, system is built by assembling and integrating existing components rather than writing code. Component integrations becomes the centerpiece in CBSD. Integrability is based on whether acquire, reuse or build the components.

Component-Based Development (CBD)[6]is a method categorizing each hardware modules and software components as a component. Component-based architectures allow selecting and loading only the necessary components for a specific application on the hardware platform.

3. Unified Model Development

Our unified model development extracts advantages from TLM, CBSD and joints them with CBD. Different abstraction levels are defined to form the construction of the development framework.

In Figure 1, we define ten system models at different abstraction levels.

The first model is Specification model. It describes the system functionality without implementation details. It is the highest level of abstraction where hardware/software (HW/SW) are described in the same context. The HW/SW are untimed functional models[7],communication are de-fined by variable-like data transfer. Figure 2 describes the sample transaction of specification model. Data foo in the top of the figure can transfer data to bar in the bottom via the variable v1. The data transfer through variable without using channel eases the conversion for C/C++ to SystemC.

The second stage is the System Partition in Figure 3. It partitions the system into HW/SW domains. [2]describes details on system partitioning. System interface and channel are also defined in this stage. The channels are untimed message passing protocols. Time is not introduced here. There-fore, wait statements are used in the code.

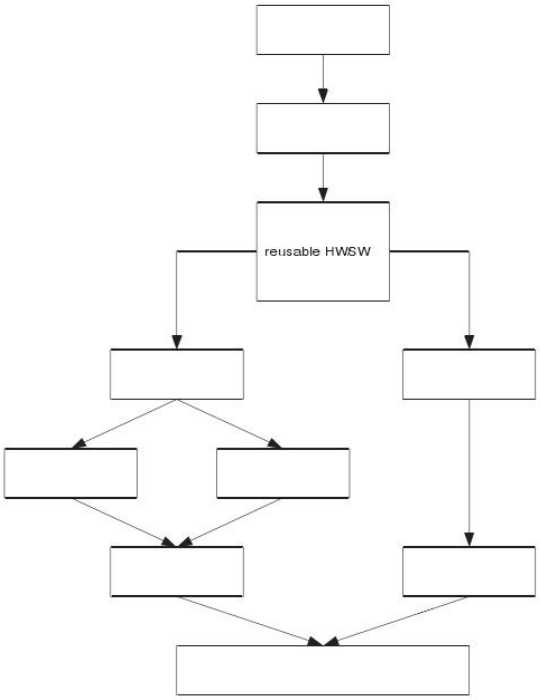

Fig. 1: Unified Component system models at different abstraction levels

The third stage is the Discovery reusable HW/SW components. In this stage, reusable components developed previously are checked with the functional models described in System Partition. The component is pulled from the library if a match or similarity is found. There are two phases of component qualification: discovery and evaluation. The functionality and interface of a component is identified in the discovery phase. In [8], evaluation process is described in particular domain. Starting from here, the system co-developed in hardware TLM and software CBSD.

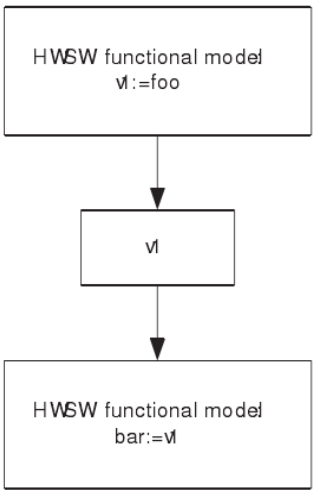

Fig. 2: Sample transaction of specification model

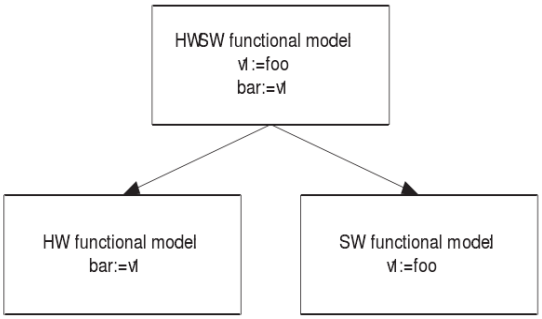

Fig. 3: System Partition model

The bus arbitration model is the first hardware TLM model. It is an abstract bus channel and implements data transfer via message protocol. Again, wait statements are used without cycle-accurate and pin-accurate details. In Figure 4, variables described in specification model can be encapsulated in the bus channel as a system bus.

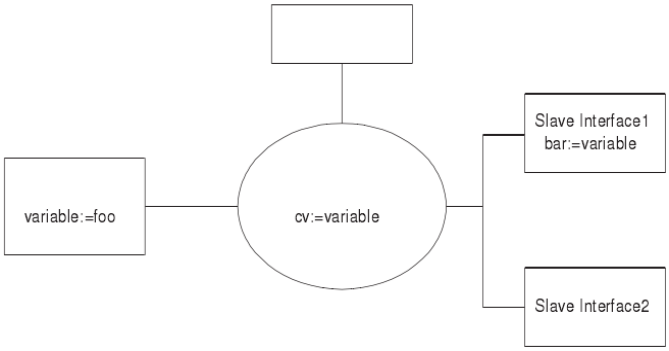

Fig. 4: Bus Arbitration model

The Time accurate and cycle accurate models specify time-accurate communication and cycle-accurate communicate. In these models, time, cycle, and pin are accurate models. Details of the state machine transaction, timings, signal sequences are described in this phase. It replaces the estimated time introduced in bus arbitration model. The result is the Implementation model is the complete model and implements in register-transfer level. It is a complete functional model.

In the Component adaption, components are written based on different assumptions in order to meet with different specifications. The adaption process must ensure that conflicts with other components are minimized. In [9], different approaches to adaption for component internal structure are described.

The assemble component process integrates components with a well-defined infrastructure. The partitioned software components can be assembled with different architectural styles such as CORBA[10].

During system integration, all components are reused or newly developed, the entire system is integrated together.Necessary adjustments are made to the component inter-faces and implementations. The system integration involves the Assemble components and Implementation model from hardware TLM and software CBSD.

4. Unified Networked Sensor Development

In this section, we show our unified networked sensor development as a example. The hardware TLMs are defined with SystemC while the software components are defined in C.

Figure 5 is our bus arbitration model. I2C cs is a chip select for the I2C bus, WE and OE are write and output en-able of the bus. It defines the input and output interface of the HW and SW. As mentioned earlier, timing is not a critical factor at this modeling level.

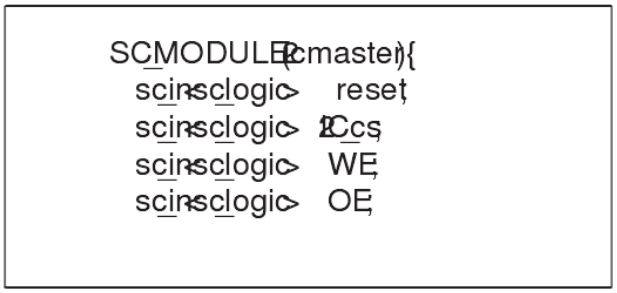

Fig. 5: I2C Bus Arbitration model

Figure 6 shows the control from the SW which can generate signals directly to SystemC. This is an example for assemble component. The rst_en allows the hardware system to be reset via the channel

TheBCTRL→Reset. The The-BCTRL→CS_WE_OE can be set with different parameters which control the chip select I2C_cs in Figure 5.

```
int main(){
  UINT rcvdata;
  TheBCTRL->Reset=rsten;
  TheBCTRL->CSWEOE=CSWE;
  rcvdata=TheBCTRL->addrdata;
```

Fig. 6: Assemble component

At Register Transfer Level (RTL), model contains details of accurate timing and cycle information. Verilog becomes a better candidate to deal with these issues. Figure 7 describes detection of acknowledgement, which is a timing and cycle accurate issue. A precise timing is critical when ack_det in is high or low and the corresponding signal assignment ack_rcv high or high impedance.

```
always @(scl or ackdeten or sclcount)begin if(str)begin
  if(ackdetin==0)begin
     rxsrin=sdain;
     ackrcv='bZ;
  end
else if((scl==0)&&(ackdeten==1)&&(sdain==0)&&
(sclcount==6'b001001))
     ackrcv=1;
```

Fig. 7: Time/Cycle accurate component

A SystemC example of implementation model is shown in Figure 8. It describes a part of the state machine within the controller of the I2C bus. In the constructor part of the SystemC, we are allowed to construct the connection between the modules. Note that in the FifoInterface, RX_fifo_emp has declared the transaction with the ProcInterface.

```
SC_CTOR(i2cmaster)
  {
     SC_METHOD(State);
     sensitive<<Clk<<Reset;
     SC_METHOD(Control);
     sensitive<<I2Ccs<<WE<<SCL_count <<ACKrcv
              <<TXfifofull<<RXfifofull;

     SC_METHOD(ProcInterface);
     sensitive<<Clk<<RXfifoemp;

     SC_METHOD(FifoInterface);
     sensitive<<Clk<<OE<<RXfifoemp;
  }
```

Fig. 8: Implementation component

The integration of the entire system is done by co-simulation using Giano[1]. Giano is a full-system simulator for embedded system. It incorporates simulation of processors, I/O systems, and the peripherals of a system. A third party Verilog interpreter with SystemC interface is attached to Giano and it is responsible for simulation of reconfigurable FPGAs. The models described above are bought to Giano to perform a complete integration process.

5. Unified Networked Sensor Development

In this paper we presented a design and modelling for a networked sensor system in a multiple-language environment. We modeled different abstraction levels in an integrated multiple-language environments where the net-worked sensor system was integrated in Giano. The system reusability and scalability are two key features of our net-worked system design. The work constituted the first experience on applying different co-design and co-simulation methodology in an embedded networked sensor system. We concluded there are advantages to use multiple languages by defining in different levels of modeling. On the other hand, we took a step further to model our communication in Matlab and Simulink. The sole purpose of using Matlab was to generate a signal spectrum in a networked environment. Therefore, by not limiting ourselves to software and hardware co-verification, we can verify the entire system by integrating with ModelSim and Matlab. Formal verification is also in progress by specifying different properties. Study has shown simulation based verification may not be adequate, we will use formal verification as an alternative way of verification in the future.

6. References

[1] A. Forin, B. Neekzad, and N. L. Lynch, ”Giano: The Two- Headed System Simulator” Microsoft Research Technical Re- port, vol. MSR-TR-2006-130, 2006.

[2] F. Fummi,M. Poncino, S. Martini, F. Ricciato, G. Perbellini, M. Turolla, ”Heterogeneous co-simulation of networked em- bedded systems” Design, Automation and Test in Europe Con- ference and Exhibition, 2004. Proceedings, vol. 3

[3] Lukai Cai and Daniel Gajski, ”Transaction Level Modeling in System Level Design” CECS Technical Report, vol. 03-10,2003

[4] D. Gajski et al., ”SpecC: Specification Language and Methodology.” Kluwer Academic Publishers, January 2000.

[5] Carnegie MellonSoftware Engineering Institute, Component-Based Softwrae Development, ”http://www.sei.cmu.edu/str/descriptions/cbsd.html”

[6] Fei Xie and James C. Browne, ”Verification of Component- Based Software Application Families”, Proc. of The 9th In- ternational SIGSOFT Symposium on Component-Based Soft- ware Engineering (CBSE 2006), 2006.

[7] Thorstn Grotker et al, ”System design with SystemC”, Kluwer Academic Publishers, 2002.

[8] Poston R.M. and Sexton M.P. ”Evaluating and Selecting Testing Tools.” IEEE Software.

[9] Valetto, G. and Kaiser, G.E. ”Enveloping Sophisticated Tools into Computer-Aided Software Engineering Environments,” Proceedings of 7th IEEE International Workshop on CASE, IEEE Computer Society Press, 1995.

[10] The Common Object Request Broker: Architecture and Specification, Version 2.0. Framingham, MA: Object Management Group, 1996.

[11] SystemC, ”http://www.systemc.org”.