SIMULATION BASED PLANNING FOR DESIGN OF WASHING MACHINE USING H/W & S/W CO DESIGN

Rajesh Kumar Garg
Lecturer,
Electronic & Communication Engineering,
Seth Jai Parkash Polytechnic, Damla,
Yamuna Nagar Haryana (INDIA)

Vikram Singh
Professor & Chairperson
Faculty of Computer Science & Applications,
Chaudhary Devi Lal University,
Sirsa, Haryana (INDIA)

ABSTRACT: A simulator is designed and developed for helping embedded system development team to plan their time schedule, work flow etc. and distribute human resources and their efforts over various phases of washing machine development. The simulator helps in identifying development phases of washing machine for successful operations of overall system. The development process of washing machine has been done in a cost effective way by planning and managing and simulating the system. The simulator gives the results in form of critical and near-critical phases while considering the rest of phases as concurrent development activities. The simulator has been a handy tool for engineers in developing the system in an optimal manner. Hardware/Software Codesign approach in which both hardware and software designers' work together to develop a system is used in this work.

KEY WORDS: Simulation, embedded system, simulator, critical index

I. INTRODUCTION

Hardware & Software Codesign*:* - Hardware and software components were developed separately in the past. Earlier the hardware designers usually make architectural choices early in the design process. These decisions were based on their knowledge of the hardware requirements and their limited knowledge and understanding of the software requirements. And they were usually hard pressed to go back and make changes to these choices. The result was that often the software designers were forced to make up for problems in the hardware through additional work of the software, often leading to a less than optimal overall design of the system. The concept of Hardware/Software Codesign is that of both hardware and software designers work together to develop a system. From specification of the requirements to exploration of the design space, and from development of the physical design to the simulation and test of the final product, hardware and software designers work cooperatively, concurrently, and most importantly, they communicate (Roundtable, 1997).

In this design methodology that has hardware and software engineers working together from the beginning of the specification phase all the way through simulation and test. In hardware/software co design, designers from both disciplines integrate their work. The process begins with a functional exploration of the project that they are undertaking. The designers define requirements and create a working specification. Then the hardware and software designers work together to map this specification on hardware and software architectures. The designers then implement these architectures onto silicon and code and come back together to simulate and test. The entire process benefits from open communication from both sides. (Garber et.al.1998) All embedded systems contain a processor and software. The processor may be micro-controller or a Pentium-IV processor. As software is there so there must be a place to store the executable code and temporary storage for run-time data manipulations. Hence ROM and RAM required. If memory requirement is small, it may be contained in the same chip as the processor. Otherwise external memory chips are used. The common features and design requirements of an embedded hardware are processing power, throughput, response, memory and power consumption, number of units, expected lifetime, program installation, testability, debug ability and reliability.

The operating system organizes and controls the hardware and it is that piece of software that turns the collection of hardware blocks into a powerful computing tool. Main tasks of the Operating system are Processor Management, Memory and Storage Management, Device Management and Providing Common Application Interface. There are four types of operating systems, based on the kind of applications are Single-user, single ; Single-user, multi-tasking ;Multi-user and Real-time operating system (RTOS).In embedded systems real time operating systems are used. The main task of a RTOS is to manage the

resources of the computer such that a particular operation executes in precisely the same amount of time every time it occur. "In a complex machine, having a part move more quickly just because system resources are available may be just as catastrophic as having it not to move at all because the system is busy."

As the complexities in the embedded applications increase, use of an operating system brings in lot of advantages. Most embedded systems also have real-time requirements demanding the use of Real time Operating Systems (RTOS) capable of meeting the embedded system requirements. Real-time Operating System allows real time applications to be designed and expanded easily. The use of an RTOS simplifies the design process by splitting the application code into separate tasks.

II. DESIGN OF SIMULATOR

A simulator is designed and developed for helping Embedded System development team, plan their time schedule, work flow etc. and distribute human resources and other efforts over various phases of washing machine development. The simulator helps in identifying development phases of washing machine which are critical and near- critical in respect of successful operations of overall system. The objective is to 1) plan, 2) organize, and 3) manage the development process more economical. The simulator gives the results in form of critical and near-critical phases while considering the rest of phases as concurrent development activities (Zeigler, 1976)The simulator will prove an asset to engineers in planning the development process optimally (Zhao et.al., 1987) Simulator is a software tool resulting from application of modeling and simulation to real life systems(Law et.al. 2000). It involves 1) modeling of system under the study i.e. washing machine 2) Representing the model in network form 3) Execution of computer form (net work form) and evaluation of simulation experiment. The aim of this research paper is to develop a simulator for planning development process of embedded system: Washing Machine. Embedded system development is complex task, is conveniently decomposed into several phases. (Ghezzi et.al. 1991) and (Pritsker, 1995) The embedded system is divided in to modules. The development of each module involves a series of activities whose temporal requirements are stochastically. Development of each phase involves carrying out various SDLC activities (Hetze, 1984) and (Payne et.al., 1989).In development of Washing Machine following life cycle activities have been considered using H/W & S/W Co Design Approach in this simulator for systematically development.

1. Design Requirements and Specifications of the embedded system
2. Hardware Selection (processor, memory, power unit and other circuits, sensors and actuators)
3. Software selection (Language, functions, tasks, ISRs)
4. Hardware Detailed Design
5. Testing of hardware
6. Software Detailed Design & Testing
7. Hardware Assembly
8. Integration and Testing

These activities have a precedence relationship amongst themselves shown in fig 1. Planning of design of washing machine is proposed in this paper is a simulator which is a software tool used in situations where analytical solution is not feasible or it is difficult to accomplish an embedded system to meet specific user requirements. The proposed simulator will act as a tool for engineers in developing washing machine for advance planning and scheduling of resources.

Network Representation of Washing Machine

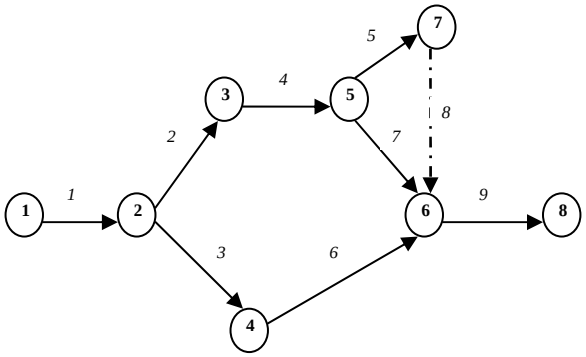

Figure 1 Network diagram of a typical washing machine

Informal Description of the Algorithm

Identifying critical activities is a 2 step process wherein during first step called forward pass the network is traversed from starting node to finishing node. This task computes the project's total completion time. Thereafter the second step called backward pass identifies the individual activities along the critical path by traversing the network starting from end node to starting node. (Narsingh, 2008)

Algorithm

1. Generate random time samples for all the activities using uniform distribution. Store them in array TIME.
2. Traverse the network in forward direction to find the length of the critical path.
3. Trace the network in backward direction to identify the activities lying on critical path.
4 Repeat the steps 2 and 3 for number of times to calculate criticality index for each activity.

i) Time duration of activities

Process of assigning duration to each activity is explained now. The time samples for activity is generated using Gaussian distribution (Box Muller transformation) as follows

$S= (-2 \log_e r1)^{0.5} * \cos(2\Pi r2)$,Where r1 and r2 are two uniform random numbers in range (0,1)

TIME [I] = σ [I] * S + μ [I], Where TIME [I] is the time sample generated randomly for activity I.

ii) Forward Pass

Equations are used during forward pass:

a) EF [I]=ES[I]+TIME[I] where I=1...N

b)EN[J]=MAX{EF[all activities terminating in J]} where J=1...M

c) ES[I]=EN[START_NODE[I]] where I=1..N

The forward pass starts with EN[I]=0. Using equation(c) we get EN[I]=ES[I]=0.

Using equation (a) we get

EF[I] = TIME[I].Proceeding this way ,by using equations (b),(c) and (a) respectively ,we reach the last node of the network and calculate the length of the critical path i.e. Tmin (total completion time of the project)

iii) Backward Pass:

Equations are used during backward pass:

a) LS[I]=LF[I]-TIME[I] where I=1..N

b)LN[J]=MIN{Ls[all activities originating in J]} where J=1..M

c)LF[every activity terminating in node J]=LN[J] where J=1..M

iv) Criticality index:

Following condition is satisfied for an activity to be a critical activity

a)LS[I]-ES[I] = LF[I]-EF[I] (latitude) is ZERO

But sometimes the condition above is not found to be exactly equal to ZERO. This is because of presence of some error. In such case the above condition is refined as follows:

LS[I]-ES[I]<=ERROR

where ERROR=0.001(assume)

Table 1 Input for simulator

ACT (K)	S [K]	F [K]	A [K]	M [K]	B [K]	μ [K]	σ [K]
1	1	2	1	2	9	3	1.78
2	2	3	2	4	12	5	2.78
3	2	4	1	4	7	4	1
4	3	5	4	6	8	6	.44
5	5	7	½	1	3/2	1	.03
6	4	6	6	8	16	9	2.78
7	5	6	5	7	15	8	2.78
8	7	6	0	0	0	0	0
9	6	8	3	5	13	6	2.78

Where

ACT[K] is ACTIVITY[K]

S [K] = START_NODE [K],

F [K] = FINISH_NODE [K],

A [K] = Optimistic estimate for each Activity K,

M [K] = Most likely duration for each activity K,

B [K] = Most pessimistic estimate for each activity K

Mean Value,

μ [K] = (A [K] +4*M [K] +B [K])/6

Standard Deviation,

σ [K] = ((B [K]-A [K])/6)

III. DISCUSSION ON RESULTS AND CONCLUSION

The Simulator was executed with the following fixed input: N=9, M=8, NRUNS 1000 ERROR=0.001. Critically Indices obtained from simulator are given in the Table2 and graph 2 shows the bar chart. Analyzing the output from the simulator output for washing machine of figure 1. It has been observed that critically indices of phases/activities baring numbers 1, 2, 4,7and 9 are on higher side than the rest of phases.

Table 2 Criticality index

Activities	Critical Index
1	0.982
2	0.909
3	0.153
4	0.912
5	0.006
6	0.153
7	0.911
8	0.006
9	1

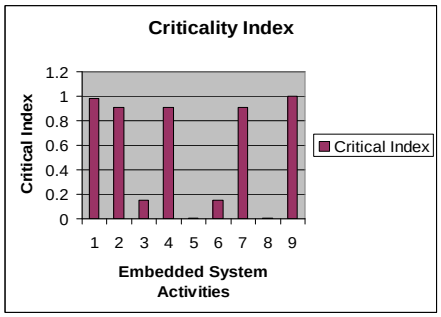

Figure 2 Criticality index bar chart

This concludes that these are more important than other phases for the planning for design of the washing machine. Any fault in these risk prone phases will result into failure. These are the activities on which more attention must be given at the time of planning for design of the washing machine. Any delay or less attention may cause a lot of loss and unnecessary delay in completion of abovementioned project.

Table 3 TMIN v/s FREQUENCY

TMIN	FRQ
15	2
16	3
17	5
18	4
19	10
20	13
21	24
22	28
23	42
24	48
25	68
26	76
27	78
28	87
29	81
30	89
31	79
32	64
33	49
34	45
35	31
36	25
37	19
38	16
39	9
40	2
43	3

The above network is simulated for 1,000 sample runs. From the graph, the minimum time for completion of the project is 15 days and maximum is 43 days. The output (i.e. the criticality index of each activity and the frequencies of occurrence of TMIN in different range) is shown in Table 2 and Table 3.

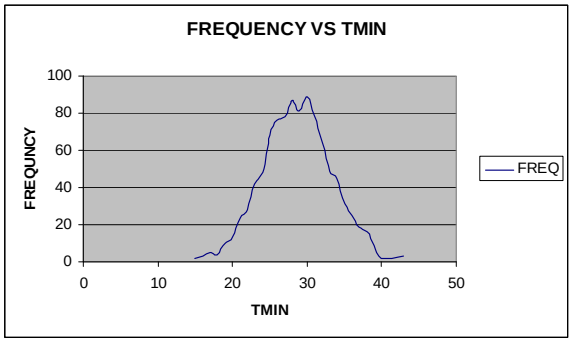

Figure 3 Frequency v/s Tmin Chart

Figure 3 shows the graph between TMIN (Project completion time) and frequency.

The graph is shown in fig 3 is appears reasonably close to normal distribution. We can use this graph as it is to estimate the probability of completing the project in a given time. Or we can approximate it to a normal distribution whose expected value and standard deviation can be determined from the data in table 3. We can also increase the number of trials from 1000 to 5000 to enhance the confidence in the graph.

IV REFERENCES

[1] L.Garber and D. Sims. (1988), "In Pursuit of Hardware-Software Co design." IEEE Computer, Vol. 31, No. 6, pp. 12-14.

[2] Noronha S. J. and Sarma V. V. S., Senior Member, IEEE, (1991) "Knowledge-Based Approaches for Scheduling Problems: A Survey", Vol. 3, No. 2, pp 160-171.

[3] R.O'Keefe, (1986) "Simulation and expert systems-A taxonomy and some examples",Vol. 46, No. 1, pp 10-16.

[4] Ramamritham, K. Stankovic, J.A., (1994) "Scheduling Algorithms and Operating Systems Support for Real-Time systems", Proceedings of the IEEE, Jan 1994, pp. 55-67

[5] Zeigler,B.P., (1976) "Theory of Modeling and Simulation," Wiley, New York.

[6] Zhao,W., et.al., (1987) "Scheduling Tasks with Resource requirement in Hard Real -Time Systems," IEEE Transactions on Software Engineering, Vol. SE-13, No.5, pp.564-577.

[7] Law, Averill M., and Kelton, W.D.,(2000) "Simulation Modeling and Analysis", 3rd ed., McGraw-Hill, New York.

[8] Ghezzi, C., Jazayeri, M. and Mandrioli, D., (1991) "Fundamentals of Software Engineering", Prentice-Hall, Englewood Cliffs, New Jersey.

[9] Pritsker, A.A. and Allen, B. (1995) "Introduction to Simulation and SLAM II", 4th ed., Wiley & Sons, New York.

[10] Hetzel W., (1984) "The Complete Guide to Software Testing", QED Information Sciences, 1984.

[11] D & T Roundtable (1987) "Hardware-Software Co design." IEEE Design and Test of Computers, Vol. 14, No.1, pp 75-83
[12] Payne, James A., (1989) “Introduction to Simulation”, McGraw-Hill – International Students Edition, Hong Kong, 1989.
[13] Deo Narsingh, (2008) “System Simulation with Digital Computer”, Eastern Economy Edition (8 ed), Prentice-Hall of India, New Delhi.