



Ain Shams University
Faculty of Engineering
Computers and System Engineering Dept.

Suboptimal Hardware/Software Codesign in Digital Systems

A THESIS

Submitted in partial fulfillment of the
requirements for the degree

Doctoral

Department of Computers and System Engineering

Ain Shams University

Submitted by

Hassan Ali Hassan Ahmed Youness

B.Sc. of Electrical Engineering
M.Sc. of Electrical Engineering
(Computers and System Engineering Dept.)
Assiut University

Supervised by:

Prof. Dr. Abd-Elmoneim Wahdan
(Ain Shams University-Egypt)

Prof. Dr. Masaharu Imai
(Osaka University-Japan)

Prof. Dr. Mohammed Moness
(Minia University-Egypt)

Prof. Dr. Ashraf Salem
(Ain Shams University-Egypt)

2009



Osaka University



Ain Shams University

Suboptimal Hardware/Software Codesign in Digital Systems

By

Hassan Ali Hassan Ahmed Youness

A THESIS

Submitted in partial fulfillment of the
requirements for the degree

Doctoral

Department of Computers and System Engineering

Ain Shams University

Cairo, Egypt.

2009

Supervised by:

Prof. Dr. Abd-Elmoneim Wahdan
(Ain Shams University-Egypt)

Prof. Dr. Masaharu Imai
(Osaka University-Japan)

Prof. Dr. Mohammed Moness
(Minia University-Egypt)

Prof. Dr. Ashraf Salem
(Ain Shams University-Egypt)

Examined by:

Prof. Dr. Dominique Borrione
(Grenoble University -France)

Prof. Dr. Ayman Wahba
(Ain Shams University-Egypt)

Prof. Dr. Abd-Elmoneim Wahdan
(Ain Shams University-Egypt)

Prof. Dr. Masaharu Imai
(Osaka University-Japan)

To my Parents, my Wife and my baby Youssef

ACKNOWLEDGMENTS

In the name of Allah, most Gracious, most Merciful

All deepest thanks are due to Almighty Allah, the merciful, the compassionate for the uncountable gifts given to me.

I am much indebted to my advisor *Prof. Dr. Imai*, Department of Information Systems Engineering, Graduate School of Information Science and Technology, Osaka University, Integrated System Design, Imai Lab, for his kind supervision, generous advice, clarifying suggestion and support during the whole scope of this work. I thank him for accepting me as his student to fulfill my dream of studying and living in Japan. In addition, special thanks to *Prof. Dr. Yoshinori Takeuchi*, and *Prof. Dr. Keishi Sakanushi*, Osaka University for their supervision, and kind assistance by supplying me with resources, and reviewing my publications, and thesis, their familiarity with the needs of the thesis was helpful in presenting this work in a professional way.

I am really indebted to my advisor *Prof. Dr. Abd-Elmonem Wahdan*, Computers and Systems Engineering Department, Ain Shams University, and my advisor *Prof. Dr. Ashraf Salem*, Computers and Systems Engineering Department, Ain Shams University, and my advisor *Prof. Dr. Mohammed Moness*, Computers and Systems Engineering, Minia University, indeed their teaching and guidance made this journey of research eventually possible.

I would like to express my great thanks to *Dr. Mohammed Abdel-Salam Hassan* for his discussions and encouragement.

I would like to thank all friends and members in the Integrated System Design Lab., at Osaka University, for their valuable cooperation that was highly needed during the conduction of this study.

A final tribute must go to my parents and my wife for whose cooperation, at all stages of this work and against all odds, have been simply overwhelming.

Hassan Ali Hassan Ahmed Youness

2009

TABLE OF CONTENTS

Dedication	ii
Acknowledgment	iii
Table of Contents	iv
List of Figures	Viii
List of Tables	xii
List of Symbols	xiv
Abstract	xvi
 Chapter 1: INTRODUCTION	 1
1.1 Overview	1
1.2 Problem Definition	4
1.3 Thesis Objectives	6
1.4 Contribution of the Thesis	7
1.5 Organization of the Thesis	9
 Chapter 2: PARALLEL SYSTEMS, MPSoCs AND PROGRAMMING	 10
2.1 Introduction	10
2.2 Flynn's Taxonomy	10
2.3 Memory Architectures	12
2.3.1 Centralized Memory	12
2.3.2 Distributed Memory	13
2.3.3 Memory Hierarchy	13
2.4 Communication Networks	14
2.4.1 Static Networks	14
2.4.1.1 Fully Connected Networks	15
2.4.1.2 Meshes	16
2.4.1.3 Hypercubes	17
2.4.2 Dynamic Networks	17
2.4.2.1 Crossbars	17
2.4.2.2 Multistage Networks	18
2.4.2.3 Tree Networks	19
2.4.2.4 Bus	19
2.5 Multiprocessor System on Chip (MPSoC)	20
2.5.1 Homogeneous MPSoC	21
2.5.2 Heterogeneous MPSoC	21
2.5.3 Application-Specific MPSoC	22
2.5.4 MPSoC Design Process	22
2.6 Parallelization	24
2.6.1 Subtask Decomposition	25
2.6.1.1 Concurrency and Granularity	26
2.6.1.2 Decomposition Techniques	26
2.6.1.3 Computation Type and Program Formulation	28
2.6.1.4 Parallelization Techniques	29
2.6.1.5 Target Parallel System	29
2.6.2 Dependence Analysis	30
2.6.2.1 Data Dependence	30
2.6.2.2 Control Dependence	31

2.7 Summary	31
Chapter 3: REVIEW AND LITERATURE SURVEY	32
3.1 Introduction	32
3.2 The DAG Model	34
3.3 Characteristics of Scheduling Algorithms	36
3.4 Classification of DAG Scheduling Algorithms	39
3.4.1 BNP Scheduling Algorithms	41
3.4.2 UNC Scheduling Algorithms	43
3.4.3 APN Scheduling Algorithms	44
3.5 Benchmark Graphs	46
3.5.1 Peer Set Graphs	46
3.5.2 Random Graphs with Optimal Solutions	46
3.5.3 Random Graphs with Pre-Determined Optimal Schedules	46
3.5.4 Random Graphs without Optimal Solutions	47
3.5.5 Traced Graphs	47
3.6 Performance Results and Comparison	48
3.6.1 Results for the Peer Set Graphs	48
3.6.2 Results for the RGBOS Benchmarks	50
3.6.3 Results for the RGPOS Benchmarks	51
3.6.4 Results for the RGNOS Benchmarks	52
3.6.4.1 Comparing Schedule Lengths	52
3.6.4.2 Number of Processors Used	54
3.6.5 Results for the Traced Graphs	55
3.6.6 Scheduling Scalability	57
3.7 Scheduling and Mapping on MPSoCs	59
3.7.1 Scheduling and Mapping in BUS Based MPSoCs	60
3.7.2 Scheduling and Mapping in NoC Based MPSoCs	62
3.7.2.1 List Heuristic Based Techniques	62
3.7.2.2 Genetic Algorithms	65
3.7.2.3 Mathematical Programming Based Techniques	66
3.7.2.4 Two Phase Scheduling Techniques	67
3.8 Hardware/Software Partitioning	67
3.8.1 Architectural Assumptions	69
3.8.2 Partitioning Objectives	70
3.8.3 Partitioning Strategies	70
3.8.4 Novel Approaches	72
3.9 Summary	76
Chapter 4: THE GEOMETRIC A* ANALYSIS SCHEDULING ALGORITHM	78
4.1 Introduction	78
4.2 A* Algorithm	79
4.3 The Proposed Algorithm	81
4.3.1 Design Principles	82
4.3.2 Task Graph Properties	87
4.3.2.1 Critical Path	88
4.3.2.2 Node Levels	88
4.3.2.3 Granularity	89
4.3.2.4 Communication-to-Computation Ratio	90

4.3.3 The GAA Algorithm	90
4.3.3.1 State-Space Search and Pruning Formulation	91
4.3.3.2 Geometric Analysis	93
4.3.3.3 The Algorithm Description	94
4.4 Illustrative Example	96
4.5 Performance Results and Comparisons	100
4.5.1 Number of Search States	100
4.5.2 Schedule Lengths	103
4.5.3 Benchmarks Comparisons	106
4.5.3.1 RGBOS	106
4.5.3.2 RGPOS	107
4.5.3.3 RGNOS	109
4.5.3.3.1 Comparing Schedule Lengths	109
4.5.3.3.2 Number of Processors Used	110
4.5.3.3.3 Scheduling Scalability	111
4.5.3.4 Large DAGs	112
4.5.3.5 Global Comparisons	115
4.6 Summary	117
Chapter 5: COMMUNICATION CONTENTION IN SCHEDULING	119
5.1 Introduction	119
5.2 Contention Wareness	120
5.2.1 End-Point Contention	121
5.2.2 Network Contention	121
5.3 Network Model	122
5.3.1 Topology Graph	122
5.3.1.1 Directed Links	122
5.3.1.2 Busses	123
5.3.1.3 Switches	123
5.4 Edge Scheduling	125
5.5 Graph Coloring Problem	126
5.6 Channel Conflict Graph (CCG)	127
5.7 Illustrative Example	128
5.8 Results and Comparisons	132
5.9 Summary	135
Chapter 6: HW/SW PARTITIONING ALGORITHM	136
6.1 Introduction	136
6.2 Target System as an Architecture Model	137
6.3 System Design Mapping Flow	138
6.4 HW-SW Partitioning Technique	139
6.5 Illustrative Example	139
6.6 Experimental Results	144
6.7 Summary	147
Chapter 7: CONCLUSIONS AND FUTURE RESEARCH DIRECTIONS	148
7.1 Conclusion	148
7.1.1 Scheduling	148
7.1.2 Communication Contention	150

7.1.3 Hardware/Software Partitioning	151
7.2 Future Research Directions	151
REFERENCES	153
PUBLICATIONS	172

LIST OF FIGURES

Figure		Page
1.1	(a) A taxonomy of the approaches to the scheduling problem; (b) A task interaction graph; (c) A task precedence graph.	5
2.1	Architecture (a) SIMD (b) MIMD.	11
2.2	Centralized memory multiprocessor.	12
2.3	Distributed memory multiprocessors (a) shared memory (b) message passing.	13
2.4	An undirected graph representing a processor network (a) with and memory associated with processor (b) processor network connection.	14
2.5	An 8 processors fully connected network.	15
2.6	Meshes.	17
2.7	Hypercubes (a) 3 dimension (b) 4 dimension.	17
2.8	An 8x8 crossbar, the source and destination units are connected through square switches.	18
2.9	A 3D butterfly network with 3 stages; the source and destination units are connected to the numbered lines.	18
2.10	Binary tree network with 8 processors.	19
2.11	An 8-processor bus topology.	20
2.12	MPSoC architecture taxonomy.	20
2.13	Homogeneous MPSoC architecture with SW-HW parts and standard programming model.	21
2.14	Heterogeneous MPSoC architecture with multiple SW system, multiple CPU models and parallel programming models.	22
2.15	Sequential programming.	25
2.16	Parallel programming – process of parallelization.	25
2.17	Data composition of matrix-vector multiplication $A.b=y$. Data is partitioned into $N=4$ parts corresponding to 4 tasks; the gray shaded part shows the data accesses by tasks 1.	27
2.18	Tree created by recursive decomposition.	27
2.19	Data dependence analysis.	30
3.1	(a) A task graph; (b) the static levels (<i>sl</i>), <i>t-level</i> , and <i>b-level</i> of the nodes.	37
3.2	Average <i>NSL</i> of the UNC, BNP and APN algorithms for RGNOS benchmarks.	53

3.3	Average <i>NSL</i> vs <i>CCR</i> .	54
3.4	The average number of processors used for the RGNOS benchmarks.	55
3.5	Average <i>NSL</i> for Cholesky factorization task graphs.	56
3.6	Average <i>NSL</i> for mean value analysis task graphs.	56
3.7	The speedups, processors scalability and running-time scalability of UNC and BNP algorithms.	58
3.8	Scheduling scalability.	59
3.9	Flow chart of Hardware/Software partitioning.	69
3.10	(a) Target architecture (b) Simple callgraph.	73
3.11	(a) System architecture (b) Dependency task graph.	73
3.12	(a) Process graph, annotated with characteristic values (b) Typical platform model.	74
3.13	The partitioning model that used by Zhang.	75
3.14	Common platform abstraction.	75
3.15	Origin (a) and modification (b) towards the common platform abstraction used for the PRES algorithm evaluation.	76
4.1	The A* algorithm.	81
4.2	DAG example (a) Task graph (b) A 3-processor target system (c) the b-level and t-level of the DAG.	83
4.3	The proposed GAA algorithm.	95
4.4	The geometrical task graph partitioned into paths and levels.	96
4.5	(a) path1 (b) path2 (c) path3.	97
4.6	(a) n_2, n_3 are equivalence nodes (b) n_2, n_3, n_4 are parallel nodes.	98
4.7	The schedule process for the task graph in Fig. 4.2 with the Gantt chart.	100
4.8	A scheduling example of one of recent work [204] for the task graph in Fig. 4.2(a) by using the A* algorithm and the proposed algorithm, they give the same schedule length. (a) The A* algorithm takes 26 states (b) Gantt chart for 3 processors (c) The GAA Algorithm takes 10 states (d) Gantt chart for 3 processors.	101
4.9	Number of States between some of the recent work and the GAA scheduling algorithms	102
4.10	The Task graph that used in comparisons for the schedule lengths among all the UNC and BNP algorithms (a) Task graph (b) Static level, top level and bottom level of the task graph.	103
4.11	The schedule of the DAG in Fig. 4.10 generated by the UNC	104

	group algorithms (a) LC (b) EZ (c) MD (d) DSC (e) DCP.	
4.12	The schedule of the DAG in Fig. 4.10 generated by the BNP group algorithms (a) HLFET (b) ISH (c) ETF (d) LAST (e) MCP (f) DLS.	105
4.13	The scheduling generated by the GAA algorithm (schedule length = 16).	106
4.14	Comparison of Average Deviation from optimal schedule length for RGBOS (UNC) algorithms and the proposed GAA at 16 processors with CCR = 0.1, 1.0 and 10.0.	107
4.15	Comparison of Average Deviation from optimal schedule length for RGBOS (BNP) algorithms and the proposed GAA at 16 processors with CCR = 0.1, 1.0 and 10.0.	107
4.16	Comparison of Average Deviation from optimal schedule length for RGPOS (UNC) algorithms and the proposed GAA at 16 processors with CCR = 0.1, 1.0 and 10.0.	108
4.17	Comparison of Average Deviation from optimal schedule length for RGPOS (BNP) algorithms and the proposed GAA at 16 processors with CCR = 0.1, 1.0 and 10.0.	108
4.18	Average NSL of the UNC algorithms for RGNOS benchmarks.	109
4.19	Average NSL of the BNP algorithms for RGNOS benchmarks.	110
4.20	Average number of processors used for the UNC algorithms for RGNOS benchmarks.	111
4.21	Average number of processors used for the BNP algorithms for RGNOS benchmarks.	111
4.22	Scheduling Scalability for the BNP RGNOS group.	112
4.23	Comparison of Average Deviation from Avg. Best schedule length for hard test graphs algorithms and the proposed GAA at 2 processors.	114
4.24	Comparison of Average Deviation from Avg. Best schedule length for hard test graphs algorithms and the proposed GAA at 4 processors.	115
4.25	Comparison of Average Deviation from Avg. Best schedule length for hard test graphs algorithms and the proposed GAA at 8 processors.	115
4.26	A global comparison of the UNC scheduling algorithms in terms of better, worse and equal performance for the RGBOS benchmarks, at CCR=10.0.	116
4.27	A global comparison of the BNP scheduling algorithms in terms of better, worse and equal performance for the RGBOS benchmarks, at CCR=10.0.	117
5.1	Representing (a) a half duplex link, (b) a full duplex link.	122

5.2	Undirected graph (a) with only point-to-point edges and hypergraph (b) with one hyperedge.	123
5.3	Networks modeled with the topology graph (P—processor, S—switch) (a) Binary tree network (b) LAN with switch (c) Switch used to model network interface bottleneck.	124
5.4	The concept of edge scheduling.	125
5.5	Conflict graph (a) CCG (b) Graph coloring table.	128
5.6	TPG example (Tasks and channels).	128
5.7	Gantt chart for Scheduling onto processors (a) one processor (P=1) (b) three processors (P=3).	129
5.8	Gantt chart for Scheduling with the external communications onto 3-processor.	130
5.9	Tasks grouped onto processors (a) all tasks are executed on one processor (b) tasks are executed on 3 processors p0, p1 and p2.	130
5.10	Conflict graph.	131
5.11	Target system.	131
5.12	Classical and New model difference in SL.	133
5.13	The general configuration for the target system.	133
5.14	Scheduling for Fig. 4.10 (chapter 4) with contention aware.	134
6.1	Target system architecture (MPSoC).	138
6.2	Design mapping flow (Scheduling + Partitioning).	138
6.3	TPG practical example.	140
6.4	Gantt chat for scheduling (a) 3 cores (b) 2 cores.	142
6.5	Scheduling and target system for 3 processors (a) task mapping and target architecture (b) Gantt chart with communication.	142
6.6	Gantt chart for task scheduling with cores optimization.	143
6.7	Implementation of tasks with communications to HW and SW (a) task mapping and target architecture (b) Gantt chart with communication.	144
6.8	HW-SW Comparisons for different task graphs for CCR = 0.1.	147

LIST OF TABLES

Table		Page
2.1	Flynn's Taxonomy.	11
2.2	Properties of static networks.	16
2.3	Properties of dynamic networks.	19
3.1	A partial taxonomy of the multiprocessor scheduling problem.	40
3.2	Some of the BNP scheduling algorithms and their characteristics.	41
3.3	Some of the UNC scheduling algorithms and their characteristics.	43
3.4	Some of the APN scheduling algorithms and their characteristics.	45
3.5	Scheduling lengths generated by the UNC and BNP algorithms for the PSGs.	49
3.6	The percentage degradations from the optimal solutions for RGBOS (UNC algorithms).	50
3.7	The percentage degradations from the optimal solutions for RGBOS (BNP algorithms).	50
3.8	The percentage degradations from the optimal solutions for RGPOS (UNC algorithms).	51
3.9	The percentage degradations from the optimal solutions for RGPOS (BNP algorithms).	52
3.10	Mapping and scheduling in BUS based MpSoC.	61
3.11	List heuristic algorithms.	65
3.12	Genetic Algorithms based scheduling.	66
3.13	Mathematical Programming based scheduling.	66
3.14	Two-phase Heuristic based scheduling.	67
3.15	Technical Characteristic of the partitioning task in several co-design environments.	72
4.1	Notation declarations.	83
4.2	Comparisons between some of recent general work and the proposed (GAA) algorithm in case of search states space.	102
4.3	Scheduling lengths generated by the UNC and BNP algorithms for Graph in Fig. 4.10.	103
4.4	Scheduling results for hard test graphs.	113
5.1	Task graph profile tables (computation costs and communication costs).	129

5.2	The results for solving TPG using classical network model.	132
5.3	The results for solving TPG using new network model.	132
6.1	Task scheduling process on 3 processors.	141
6.2	Task scheduling process on 2 processors.	141
6.3	Schedule lengths for RGBOS at CCR = 0.1, 1.0 and 10.0 using 1-to-8 processors.	145
6.4	Schedule lengths for hard task graphs at $\rho = 60$ and 80.	146
6.5	Comparing number of channels and Buses for CCR=0.1 of RGBOS benchmark.	146

LIST OF SYMBOLS

Symbol	Declaration
$G(V,E,w,c)$	Task graph (DAG) Directed Acyclic Graph
SL	Schedule Length
sl	Static Level (b-level without edge costs)
bl	Bottom level(the length of the longest path from a node to an exit node)
tl	Top level (the length of the longest path from an entry node to the node)
cp	Critical path (the longest path in the graph)
n	Task number
V	Number of tasks in the task graph
E	Number of edges in the task graph
P	Number of processors in the target system
P_k	Processor number
P_{rt}	Ready time of processor
P_{src}	Source processor
P_{dst}	Destination processor
$Paths$	Number of paths in the task graph
$Path$	The path number
$Levels$	Number of levels in the task graph
$Level$	The level number
CCR	The communication-to-computation ratio
$ST(n_i)$	Start time of node n_i
$FT(n_i)$	Finish time of node n_i
$Pred(n_i)$	Set of predecessors of node n_i