Hardware RMS Scheduler

Thomas Gaul, Jackson Hafele CPR E 558 Final Project Fall 2023

IOWA STATE UNIVERSITY

Problem Statement

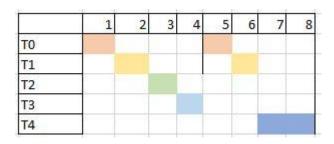
- Software implementation of task schedulers can require large amounts of overhead
- May not be ideal of low level embedded systems application with limited amounts of resources
- Alternative is to implement a hardware scheduler with either FPGA or ASIC for faster context switching or lower amount of power consumption

Solution

Schedule Implementation

- Implement static RMS Schedule
- Tasks refresh based on static period (Pi)
- Tasks take N time units to complete based on computation time (Ci)
- Highest priority goes to task with smallest period
- Sample task set created to model on Hardware

Task	Period (Pi)	Computation Time (Ci)			
TO	4	1			
T1	4	1			
T2	8	1			
T3	8	1			
T4	8	2			

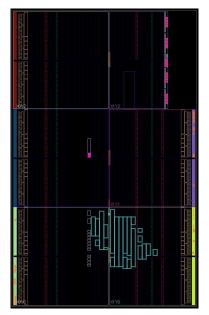


IOWA STATE UNIVERSITY

Solution

Hardware Application

- Model RMS Schedule on Hardware with fixed parameters
- Repeat indefinitely while running periodic task set
- To create
 - Implement register to store data of each task
 - Implement control module for task context switching
 - Implement decoding between tasks based on completion and priority

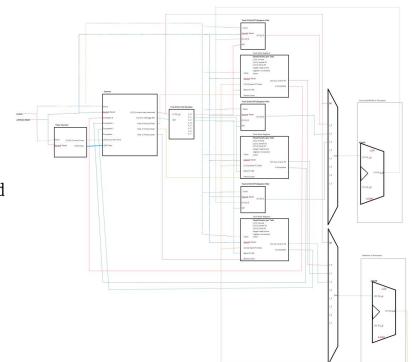


FPGA Utilization Report

IOWA STATE UNIVERSITY

Implementation details

- Tools
 - Verilog/VHDL
 - ModelSim on the Coover Linux VDI servers
 - Vivado
 - Xilinx FPGA Arty Basys 3 Development Board



IOWA STATE UNIVERSITY

Implementation details

• Time Counter

- tracks of elapsed time to ensure tasks are reset at the correct deadline
- Control
 - Selects which task to be active with RMS scheduling based on the period and completion status

• 3 to 8 One Hot Decoder

- Controls the write enable for each tasks register file and task state register
- Task Register File (32bit DFF)
 - Each task has its own register file that interface with the ALU to allow for fast context switching
- Task State Register
 - Tracks each task in terms of completion and its program counter and resets to the beginning of the task at the beginning of its period

IOWA STATE UNIVERSITY

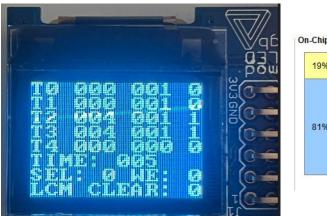
Testing/Evaluation Results

Constants	-																		_
	-No Data-	3									3								
	-No Data-	3																	
TASK2PERIOD	-No Data-	7																	
	-No Data-	7																	
	-No Data-	7																	
TASKO INITIAL PC	-No Data-	0																	
TASK1 INITIAL PC	-No Data-	0																	
TASK2 INITIAL PC	-No Data-	0																	
TASK3 INITIAL PC	-No Data-	0																	
	-No Data-	0																	
TASKO FINAL PC	-No Data-	4																	
TASK1 FINAL PC	-No Data-	4																	
TASK2 FINAL PC	-No Data-	4																	
TASK3 FINAL PC	-No Data-	4																	
TASK4 FINAL PC	-No Data-	8																	
- Standard Inputs		_																	
s CLK	-No Data-																		
- Control Outputs																			
• o_time	-No Data-	(0	Ύ1	12	13	4	15	Ĭ6	17	χo	1	12	13	14	15	16	17	Io	Y 1
o LCMclear	-No Data-									_									
• o Current Task Sel	-No Data-	(7	ĬO	11	2	[3	lo	11	4		ĬO	11	2	13	10	1	4		ĬO
- Task State Outputs		_																	
🕶 🔶 o task period clear	-No Data-	(11111	00000			00011	00000			11111	00000			00011	00000			11111	00000
• o task isComplete	-No Data-	00000		00001	00011		01100	01101	01111		00000		00011	00111	01100	01101	01111		00000
— Task Current PC Outputs —		14																	
• o task0 currentPC	-No Data-	0		4			10	4			10	4			lo l	4			0
• o task1 currentPC	-No Data-	(0			4		X0		4		10		4		(o		(4		0
• o task2 currentPC	-No Data-	0				4					10			4					0
• o task3 currentPC	-No Data-	0					<u> </u> 4				0				4				0
• o task4 currentPC	-No Data-	0								4	0							4	0
🗉 🔶 s Task0 Deadline	-No Data-	3				7				3				7				3	
🗉 🧄 s Task1 Deadline	-No Data-	3				7				3				7				3	
• s Task2 Deadline	-No Data-	(7																	
• s Task3 Deadline	-No Data-	(7																	
s Task4 Deadline	-No Data-	(7																	
🕒 🤝 s_Task4_Deadline	-No Data-																		

IOWA STATE UNIVERSITY

Testing/Evaluation Results

- Advantages:
 - No OS overhead(64 cycles + for context Switching)
- Cost (Vivado)
 - Worst Negative Slack:3.937ns
 - Total Power Consumption:0.089W
 - Look Up Tables Used: 1120
 - Flip Flops Used: 1122



19%	Dynamic	:: 0.0	17 W (199	6)
81%	8%	Clocks:	0.001 W	(5%)
	8%	Signals:	0.001 W	(8%)
	79%	Logic:	0.001 W	(8%)
	1	1/0:	0.013 W	(79%)

IOWA STATE UNIVERSITY

Conclusions

- Successfully modeled sample RMS schedule with waveform results and FPGA synthesis
- Applied topics from related courses such as CPRE 281, CPRE 381, and CPRE 581
- Created platform for further work with other real-time schedulers or dynamic properties

• Future Work

- EDF Implementation
- Aperiodic Events
- Integration with full processor

IOWA STATE UNIVERSITY

Learning Achieved through the project

- Applied RMS schedule to hands-on FPGA application
- Used real-time systems topics in a unique fashion
- Utilized experience gained from other digital design classes for FPGA application in new context
- Strengthened hardware design language skills
- Learned about research in the area of hardware for real time systems

IOWA STATE UNIVERSITY