Partial Region and Bitstream Cost Models for Hardware Multitasking on Partially Reconfigurable FPGAs

Aurelio Morales-Villanueva and Ann Gordon-Ross
NSF Center for High-Performance Reconfigurable Computing (CHREC)
Department of Electrical and Computer Engineering, University of Florida
Gainesville, Florida, USA
E-mail: {morales, ann}@chrec.org

Abstract—Partial reconfiguration (PR) on field-programmable gate arrays (FPGAs) enables multiple PR modules (PRMs) to time multiplex partially reconfigurable regions (PRRs), which affords reduced reconfiguration time, area overhead, etc., as compared to non-PR systems. However, to effectively leverage PR, system designers must determine appropriate PRR sizes/organizations during early stages of PR system design, since inappropriate PRRs, given PRM requirements, can negate PR benefits, potentially resulting in system performance worse than a functionally-equivalent non-PR design. To aid in PR system design, we present two portable, high-level cost models, which are based on the synthesis report results generated by Xilinx tools. These cost models estimate PRR size/organization given the PRR's associated PRMs to maximize the PRRs' resource utilizations and estimate the PRM's associated partial bitstream sizes based on the PRR sizes/organizations. Experiments evaluate our cost models' accuracies for different PRMs and required resources, which enable our models to afford enhanced designer productivity since these models preclude the lengthy PR design flow, which is typically required to attain such analysis.

Keywords—FPGA, partial reconfiguration, cost model, hardware multitasking.

I. Introduction and Motivation

Partial reconfiguration (PR) on field-programmable gate arrays (FPGAs) partitions the FPGA fabric into one static region and one or more partially reconfigurable regions (PRRs), which enables PRRs to time-multiplex multiple hardware tasks (i.e., PR modules (PRMs)). A PRR is reconfigured using the PRM's partial bitstream, which contains the PRM's flip-flops' (FFs') and memory blocks' initial values. Since this partial bitstream only reconfigures one PRR, as compared to full reconfiguration of the entire FPGA using a full bitstream, PR affords faster reconfiguration time and smaller bitstreams. Unlike full reconfiguration that halts the entire FPGA's execution, PRR reconfiguration isolates the reconfiguration to a single PRR without halting the static region's or other PRRs' executions.

This isolated reconfiguration and hardware multitasking of PRMs provides additional PR benefits as compared to full reconfiguration, such as reduced FPGA area requirements and power consumption [3][8]. Furthermore, PRR reconfiguration is flexible and can be executed dynamically using either the internal configuration access port (ICAP) on the FPGA, or an external controller, such as a host PC, but using the ICAP offers the advantage of autonomous system reconfiguration after system deployment.

However, effectively leveraging PR benefits is challenging for system designers. Many design decisions affect overall PR system performance, and if not adequately chosen, inappropriate decisions can result in severe adverse effects, potentially leading to PR system performance that is worse than a non-PR system. For example, even though large PRRs afford greater PRM time-multiplexing potential (i.e., large PRRs contain more computational resources than small PRRs and can thus accommodate a wider variety of functionality/PRMs), oversized PRRs impose longer routing delays and reconfiguration time, reduced parallelization potential due to fewer PRRs, and thus potentially worse performance than a non-PR system.

Selecting an appropriate PRR size (total number of rows and columns in the PRR) and organization (specific resources distributed in the PRR) is a critical design decision that must be done early in system design, during design partitioning at the system and application levels, in order to increase designer productivity. The PRRs' and static region's sizes are dictated by which partitions (i.e., PRMs) of the application are assigned to these regions. When selecting PRMs to time-multiplex PRRs, the system designer must partition the application into multiple PRMs and consider the intended time-multiplexed PRMs' different resource requirements (e.g., configurable logic blocks (CLBs), digital signal processing blocks (DSPs), and random access memory blocks (BRAMs)), which dictates the PRR's size and organization (e.g., resource distribution). Since PR partitioning can range from the finest grained, where PRMs are defined on a per-instruction/operation basis, to the coarsest grained, where the entire application is defined as a single PRM, and each partitioning between these ranges offers design tradeoffs (e.g., power, parallelization potential, performance, device size, etc.), the PR partitioning design space is exponentially large and designers can only feasibly evaluate a subset of these designs.

To assist in early PR partitioning design decisions, system designers need system/application-level analytical or simulated models that evaluate the impact of these decisions on the PRR size/organization and bitstream sizes. Without these high-level models to determine these impacts, for every considered PR partitioning, the system designer must perform complete PR system implementation, which includes iterative execution of the lengthy PR design flow (e.g., design creation and synthesis of all PRMs and the static region, manual PRR floorplanning, place and route of the static region and PRRs for the PRMs, generation of the full and PRMs' partial bitstreams, etc.). Even though complete implementation provides highly accurate design analysis and PR benefits, the design time effort may be prohibitive since this PR design flow can take hours to days, depending on the system complexity, to implement a single PR partitioning. Therefore, designers must have high-level cost models that quickly evaluate design decisions early in the design process and provide sufficiently accurate evaluation, which significantly reduces design space exploration time as compare to full system implementation. These cost models evaluate the impact of PRR size/organization selection with respect to the maximum resource utilization for the PRRs' associated PRMs, which in turn affects the PRMs' associated partial bitstream sizes.

Some prior research develops and evaluates cost models for PR FPGAs, but these prior works did not present a holistic method for evaluating the tradeoffs considering the PRR size/organization's impact on partial bitstream size, reconfiguration time, and overall PR system performance, which are necessary for a complete and holistic assessment of PR design decisions. Thus we propose two high-level cost models based on the synthesis report results generated by Xilinx tools. These models use mathematical formulas to determine the PRR's sizes/organizations and partial bitstream sizes considering designer-defined multitasking PRMs (i.e., our work is based on designer-defined PRM design decisions, since PRM definition is beyond the scope of this research) and the PRMs' associated partial bitstream sizes.

In this paper, we introduce the first, to the best of our knowledge, detailed cost model for determining partial bitstream sizes without executing the entire PR design flow (no other prior works or vendor tool documentation provide this analysis). We define our cost models to be generally portable across different Xilinx FPGA families by simply altering the cost model's device-specific characteristics' values in the cost models' formulas. We show the efficacy of our PRR size/organization cost model using complex PRMs, and compare the model's estimated PRR sizes/organizations using synthesis reports against the PRR sizes/organizations obtained by executing the entire PR design flow. Our cost models' ability to estimate the PRR size/organization and the PRMs' associated partial bitstream size without completing the entire PR design flow significantly decreases design exploration time and, thus increases designer productivity and adherence to design goals (e.g., performance and power constraints, etc.).

II. Related Work

Prior works in PR cost models only provided partial methods for evaluating design tradeoffs. Liu et al. [4] compared multiple PR designs using the ICAP and proposed a direct memory access (DMA)-based PR design to reduce the PRR reconfiguration time. The authors compared the reconfiguration time for the different PR designs and different bitstream sizes, but the results did not include details about the PRRs' sizes/organizations.

Papadimitriou et al. [7] presented an extensive survey of PRR reconfiguration times and introduced a cost model for PRR reconfiguration based on the bitstream storage media type (e.g., compact flash, BRAM, DDR SDRAM, etc.). However, the cost model's estimation had a 30% to 60% error as compared to the measured reconfiguration times. Claus et al. [1] used formulas to calculate the expected PRR reconfiguration time based on the ICAP's busy-factor, which reflects the ICAP's shared resource contention for PRR reconfiguration. However, since this approach only considered the ICAP's busy-factor, the method is only valid if the ICAP is the limiting factor during reconfiguration. Furthermore, the method was only applicable to the Virtex-II FPGAs, and was not portable to different FPGA families.

Duhem et al. [2] introduced FaRM, a high-speed internal configuration controller for Xilinx FPGAs, and presented a cost model to evaluate the PRR reconfiguration time, but the authors did not verify the cost model with measured values, and did not provide reconfiguration time analysis for different partial bitstream sizes.

Although all of these prior works analyzed certain PR design decisions, none considered holistic PR design evaluation and exploration of the PRRs' sizes/organizations and how the PRRs' sizes/organizations affects the partial bitstream size, which in turn also affects the reconfiguration time and overall PR system performance. In prior work [5][6], we used the proposed cost models (only for CLBs) as part of design space exploration for two PR designs for fine-grain evaluation of PRR reconfiguration times in two hardware multitasking applications. Since our prior work only considered CLB resources, this paper presents significant cost model enhancements that include CLBs, DSPs, and BRAMs, and presents the cost models' portability across FPGA families.

III. Cost Models for PR FPGAs

Our proposed cost models for PR designs on Xilinx FPGAs assist system designers in quickly evaluating and selecting the PRRs' sizes/organizations that minimize partial bitstream size and reconfiguration time, with respect to design goals, without executing the entire PR design flow, thereby increasing designer productivity. Since low-level device details are critical to cost model estimation accuracy and general understanding for portability to other device families, we present a brief description of key aspects of the Virtex-5 device family. Using this knowledge foundation, we establish our cost model formulas for PRR size/organization based on the PRMs' required resources, and present the formulas for partial bitstream size derivation, which are based on the PRR size/organization.

A. Xilinx Virtex-5 Device Layout and Resources

The Xilinx Virtex-5 FPGA family and newer families, such as the Virtex-6 and -7 series and the Zynq-7000, support two-dimensional PR, which allows PRRs to have any rectangular shape on the device fabric, enabling PRRs to contain a diverse mixture of resources based on the PRR's fabric location. The FPGA's resources are distributed into a row/column

Table I. PARAMETERS USED IN THE PRR SIZE/ORGANIZATION COST MODEL

Parameter	Description
LUT_FF_{req}	LUT FF pairs required in PRM
LUT_{req}	Slice LUTs required in PRM
LUT_CLB	LUTs per CLB
FF_CLB	FFs per CLB
CLB_{req}	CLBs required in PRM
FF_{req}	FFs required in PRM
W_{CLB}	CLB columns in PRR
H_{CLB}	CLB rows in PRR
CLB_{col}	CLBs in a column (per row)
DSP_{req}	DSPs required in PRM
W_{DSP}	DSP columns in PRR
H_{DSP}	DSP rows in PRR
DSP_{col}	DSPs in a column (per row)
$BRAM_{req}$	BRAMs required in PRM
W_{BRAM}	BRAM columns in PRR
H_{BRAM}	BRAM rows in PRR
$BRAM_{col}$	BRAMs in a column (per row)
CLB_{avail}	CLBs available in PRR
FF_{avail}	FFs available in PRR
DSP_{avail}	DSPs available in PRR
$BRAM_{avail}$	BRAMs available in PRR
H	Number of rows in the PRR
W	Number of columns in the PRR
PRR_{size}	Size of PRR

organization where each column contains a group of configuration frames and the number of configuration frames per column depends on the resource type (e.g., CLB, DSP, etc.). A frame is the minimum unit of information used to configure/read the FFs' stored values and BRAMs in the device's configuration memory (CM). Input/output blocks (IOBs) and clock (CLK) resources are not supported as part of the PRRs in the current versions of the Xilinx tools.

For Virtex-5 devices, a frame contains 41 32-bit words, and CLB, DSP, BRAM, IOB, and CLK columns have 36, 28, 30, 54, and 4 configuration frames, respectively. Each BRAM column requires 128 data frames for BRAM initialization. In any given row, a CLB column has 20 CLBs, a DSP column has 8 DSPs, and a BRAM column has 4 BRAMs. Each CLB contains a pair of slices and each slice contains 4 look-up tables (LUTs) and 4 FFs. We refer the reader to [9][10] for a complete description of the Virtex-5 device architecture and configuration.

B. Cost Model for PRR Size/Organization

The PRR size/organization affect the partial bitstream size. Since the PRR floorplanning in the PR design flow is a manual process executed by system designers, oversized PRRs or ill-suited PRR resource row/column organization with respect to the associated PRMs' resource requirements can increase the internal fragmentation (PRR resources not used in the associated PRM), which unnecessarily increases the PRR's number of configuration frames, partial bitstream size, and reconfiguration time. Our cost model assists system designers in determining the PRR size/organization given the PRR's associated PRMs in order to reduce internal fragmentation and without completing the entire PR design flow.

Table I defines the parameters used in the PRR size/organization cost model for PRMs containing CLBs, DSPs, and BRAMs, where LUT_CLB, FF_CLB, CLB_{col}, DSP_{col}, and $BRAM_{col}$ are device-family dependent. The PRM's resources can be obtained from the synthesis report generated by synthesizing the PRM using the Xilinx Synthesis Technology (XST) tool to obtain the parameters LUT_FF_{req}, LUT_{req}, FF_{req}, DSP_{req}, and $BRAM_{req}$. These parameters are sufficient to determine the PRR size/organization of the associated PRM. Table II summarizes the Virtex-4/-5/-6 device family's specific values for CLB_{col}, DSP_{col}, $BRAM_{col}$, LUT_CLB, and FF_CLB, used in Table I.

Table II. SPECIFIC VALUES FROM TABLE I FOR VIRTEX-4, -5, AND -6 DEVICE FAMILIES

Parameter	Virtex-4	Virtex-5	Virtex-6
CLB_{col}	16	20	40
DSP_{col}	4	8	16
$BRAM_{col}$	4	4	8
LUT_CLB	8	8	8
FF_CLB	8	8	16

The LUT_FF_{req} parameter represents the PRM's required number of LUTs paired with one FF within a slice, where LUT_FF_{req} includes: the LUT FF pairs with unused LUTs (only FFs), the LUT FF pairs with full use of the LUT FF pairs, and the LUT FF pairs with unused FFs (only LUTs). FF_{req} is the addition of both the LUT FF pairs with unused LUTs and with full use of the LUT FF pairs. LUT_{req} is the addition of both the LUT FF pairs with full use of the LUT FF pairs and with

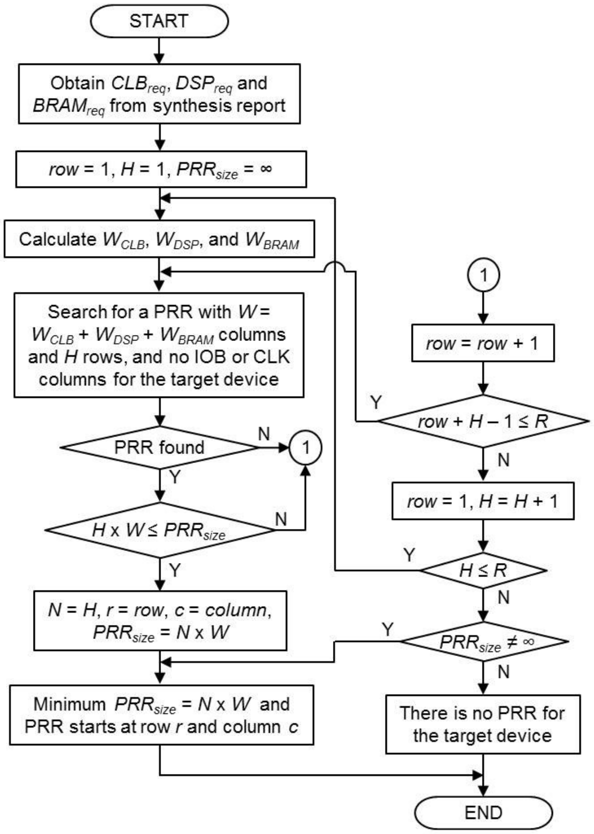
START
Obtain CLB_{req}, DSP_{req} and $BRAM_{req}$ from synthesis report
row = 1, H = 1, PRR_{size} = ∞
Calculate W_{CLB}, W_{DSP}, and W_{BRAM}
Search for a PRR with W = W_{CLB} + W_{DSP} + W_{BRAM} columns and H rows, and no IOB or CLK columns for the target device
PRR found
N
Y
1
H x W ≤ PRR_{size}
N
Y
N = H, r = row, c = $column$, PRR_{size} = N x W
Minimum PRR_{size} = N x W and PRR starts at row r and column c
1
row = row + 1
Y
row + H – 1 ≤ R
N
row = 1, H = H + 1
Y
H ≤ R
N
Y
PRR_{size} ≠ ∞
N
There is no PRR for the target device
END

Fig. 1. Flow to obtain the PRR size/organization from the required resources in the synthesis report

unused FFs.

The number of CLBs required in the PRM (CLB_{req}) is:

$$CLB_{req} = \lceil LUT\_FF_{req}/LUT\_CLB \rceil \quad (1)$$

Since the LUT_FF_{req}/LUT_CLB may be a non-integer, we take the ceiling of this value to ensure sufficient CLB resources. The number of CLB columns in a PRR (W_{CLB}) for H_{CLB} rows in the PRR using (1) is:

$$W_{CLB} = \lceil CLB_{req}/(H_{CLB} \times CLB_{col}) \rceil \quad (2)$$

The number of DSP columns in a PRR (W_{DSP}) for H_{DSP} rows in the PRR can be obtained from the number of DSPs required in the PRM (DSP_{req}) as:

$$W_{DSP} = \lceil DSP_{req}/(H_{DSP} \times DSP_{col}) \rceil \quad (3)$$

We note that some Xilinx devices include only one DSP column in the fabric, which sets $W_{DSP} = 1$ in (3). In these cases the number of DSP rows (H_{DSP}) in the PRR is:

$$H_{DSP} = \lceil DSP_{req}/(W_{DSP} \times DSP_{col}) \rceil \quad (4)$$

The number of BRAM columns in a PRR (W_{BRAM}) for H_{BRAM} rows in the PRR is determined using the PRM's required number of BRAMs ($BRAM_{req}$) as:

$$W_{BRAM} = \lceil BRAM_{req}/(H_{BRAM} \times BRAM_{col}) \rceil \quad (5)$$

For a rectangular PRR, $H_{CLB} = H_{DSP} = H_{BRAM} = H$, and the total number of columns in the PRR (W) is:

$$W = W_{CLB} + W_{DSP} + W_{BRAM} \quad (6)$$

The PRR's size (PRR_{size}) is:

$$PRR_{size} = H \times W \quad (7)$$

For multiple PRMs that share the same PRR, each PRM has a unique H, and the largest W_{CLB}, W_{DSP}, and W_{BRAM} across all of the PRR's associated PRMs dictates the number of CLB, DSP, and BRAM columns in the PRR, respectively.

Internal fragmentation occurs when the expressions within the ceiling functions of (1), (2), (3) (or (4)), and (5) are non-integers. To reduce the internal fragmentation for CLBs, DSPs, and BRAMs, the expressions within the ceiling functions of (2), (3) (or (4)), and (5) should be as close to the next highest integer as possible.

According to (7), the PRR should have H rows and W columns, but we need to determine if it is possible to find a physical area in the target device that satisfies this condition. Fig. 1 depicts the flow to search for and obtain the PRR size/organization that satisfies (1) to (7) using the parameters CLB_{req}, DSP_{req}, and $BRAM_{req}$ from the synthesis report for the PRM and the selected target device. In order to produce the lowest internal fragmentation and lowest partial bitstream size for a PRM, H should start at $H = 1$ and verify if it is possible to distribute the CLBs, DSPs, and BRAMs in W contiguous columns (no IOB or CLK columns in the PRR) using (2) to (6) for the target device. The search for a PRR starts at the bottom of the device fabric ($row = 1$), where the target device has R rows. Internal fragmentation occurs if it is not possible to distribute the CLBs, DSPs, and BRAMs in W contiguous columns (distributing the CLB, DSP, and BRAM columns in any order) for a given H and for the target device.

Table III. PARAMETERS USED IN THE BITSTREAM SIZE COST MODEL

Parameter	Description
IW	Number of initial words
FW	Number of final words
FAR_FDRI	FAR/FDRI initialization words per row
NCW_{row}	Configuration words in a PRR row
NDW_{BRAM}	BRAM initialization words in a PRR row
NCF_{CLB}	CLB configuration frames in a PRR row
NCF_{DSP}	DSP configuration frames in a PRR row
NCF_{BRAM}	BRAM configuration frames in a PRR row
CF_{CLB}	Configuration frames per CLB column
CF_{DSP}	Configuration frames per DSP column
CF_{BRAM}	Configuration frames per BRAM col.
DF_{BRAM}	Initialization frames per BRAM col.
FR_{size}	Frame size in words
$Bytes_{word}$	Number of bytes per word
H	Number of rows in the PRR
$S_{bitstream}$	Size of partial bitstream in bytes

If it is not possible to find a PRR for the current H, H is incremented and W_{CLB}, W_{DSP} (or H_{DSP}), and W_{BRAM} in (2), (3) (or (4)), and (5), respectively, are recalculated and the search for the PRR starts again from the bottom of the device fabric. Once a PRR is found, the PRR position in the device fabric will be at row r and column c, where $r + H - 1 \le R$, and c is the left most column position of the PRR in the device fabric.

From Table I, the CLBs, FFs, LUTs, DSPs, and BRAMs available in the PRR are:

$$CLB_{avail} = H_{CLB} \times W_{CLB} \times CLB_{col} \quad (8)$$

$$FF_{avail} = CLB_{avail} \times FF\_CLB \quad (9)$$

$$LUT_{avail} = CLB_{avail} \times LUT\_CLB \quad (10)$$

$$DSP_{avail} = H_{DSP} \times W_{DSP} \times DSP_{col} \quad (11)$$

$$BRAM_{avail} = H_{BRAM} \times W_{BRAM} \times BRAM_{col} \quad (12)$$

Internal fragmentation is dictated by the PRR's resource utilization (RU). RU is the percentage of the resources used by

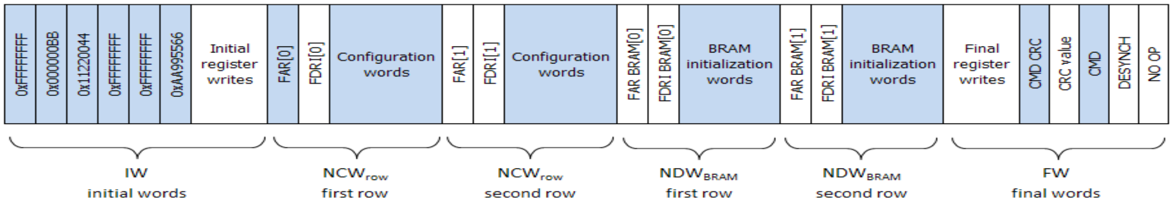

Fig. 2. Partial bitstream structure for Virtex-5 FPGAs

Table IV. SPECIFIC VALUES FROM TABLE III FOR VIRTEX-4, -5, AND -6 DEVICE FAMILIES

Parameter	Virtex-4	Virtex-5	Virtex-6
CF_{CLB}	22	36	36
CF_{DSP}	21	28	28
CF_{BRAM}	20	30	28
DF_{BRAM}	64	128	128
FR_{size}	41	41	81
IW	12	16	20
FW	108	114	113
FAR_FDRI	5	5	5
$Bytes_{word}$	4	4	4

the PRR's associated PRMs compared to the PRR's available resources, wherein a high *RU* means a low internal fragmentation. *RU* is measured for each resource type as:

$$RU_{CLB} = \left(CLB_{req}/CLB_{avail}\right) \times 100\% \tag{13}$$

$$RU_{FF} = \left(FF_{req}/FF_{avail}\right) \times 100\% \tag{14}$$

$$RU_{LUT} = \left(LUT_{req}/LUT_{avail}\right) \times 100\% \tag{15}$$

$$RU_{DSP} = \left(DSP_{req}/DSP_{avail}\right) \times 100\% \tag{16}$$

$$RU_{BRAM} = \left(BRAM_{req}/BRAM_{avail}\right) \times 100\% \tag{17}$$

C. Cost Model for Partial Bitstream Size

The partial bitstream size can be calculated using the PRR size/organization formulas (Section III.B) and details of the partial bitstream organization. No prior work or technical documents from Xilinx specify how to exactly obtain the partial bitstream size of a PRR, and specifically without executing the entire PR design flow. Our proposed cost model assists system designers in early PR design decisions in the absence of this information.

Table V. APPLICATION OF OUR PRR SIZE/ORGANIZATION COST MODEL FOR VIRTEX-5 AND -6 DEVICE FAMILIES

	Virtex-5 LX110T			Virtex-6 LX75T		
Parameter	FIR	MIPS	SDRAM	FIR	MIPS	SDRAM
LUT_FF_{req}	1300	2619	332	1466	3238	385
DSP_{req}	32	4	0	27	4	0
$BRAM_{req}$	0	6	0	0	6	0
LUT_{req}	1150	1527	157	1317	2096	181
FF_{req}	394	1592	292	394	1860	324
CLB_{req}	163	328	42	184	405	49
H_{CLB}	5	1	1	1	1	1
W_{CLB}	2	17	3	5	11	2
H_{DSP}	5	1	0	1	1	0
W_{DSP}	1	1	0	2	1	0
H_{BRAM}	0	1	0	0	1	0
W_{BRAM}	0	2	0	0	1	0
CLB_{avail}	200	340	60	200	440	80
FF_{avail}	1600	2720	480	3200	7040	1280
LUT_{avail}	1600	2720	480	1600	3520	640
DSP_{avail}	40	8	0	32	16	0
$BRAM_{avail}$	0	8	0	0	8	0
RU_{CLB}	82%	97%	70%	92%	92%	61%
RU_{FF}	25%	59%	61%	12%	26%	25%
RU_{LUT}	72%	56%	33%	82%	60%	28%
RU_{DSP}	80%	50%	0%	84%	25%	0%
RU_{BRAM}	0%	75%	0%	0%	75%	0%

Fig. 2 depicts the Virtex-5 partial bitstream structure, which is similar across device families. The partial bitstream is composed of a set of initial words, followed by a set of configuration words, BRAM initialization words (if the PRR contains BRAMs), and a set of final words. Fig. 2 depicts a sample partial bitstream structure for a PRR with two rows that contain CLBs, DSPs, and BRAMs. From this bitstream, we remove the initial bytes, including the name of the native circuit description file (*.ncd) used to generate the partial bitstream and the bitstream creation date, resulting in a 32-bit word aligned bitstream. The initial/final words in the partial bitstream are used for synchronization/desynchronization of the bitstream with the ICAP. Synchronization alerts the device of upcoming configuration words, and aligns the configuration words with the internal configuration logic, and desynchronization releases the ICAP, which allows other PRRs to be reconfigured. We refer the reader to [9] for a detailed description of the partial bitstream structure for Virtex-5 FPGAs.

Table III depicts the parameters for partial bitstream size derivation where IW, FW, FAR_FDRI, CF_{CLB}, CF_{DSP}, CF_{BRAM}, DF_{BRAM}, and FR_{size} are device family dependent. We note that for Virtex-4/5/6 and Series 7 devices, words are 32-bit, however, in other devices, such as Spartan-3/6 devices, words are 16-bit, therefore, $Bytes_{word}$ must be adjusted according to the device family. In Table III, the FAR_FDRI specifies the number of words for setting the frame address register (FAR) and the frame data register input (FDRI) register [9]. The FAR specifies the first frame address in terms of a row and column on the device fabric for configuration words (or initialization words for BRAM columns, if BRAMs are used) in a given PRR row, and the FDRI specifies the number of configuration words (or initialization words for BRAM columns) for the given PRR row. Table IV summarizes the specific values from Table III for Virtex-4/5/6 device families.

The size of the partial bitstream ($S_{bitstream}$) for a PRR with H rows that contains CLBs, DSPs, and BRAMs is:

$$S_{bitstream} = \{IW + H \times (NCW_{row} + NDW_{BRAM}) + FW\} \times Bytes_{word} \tag{18}$$

The number of configuration words in a PRR row (NCW_{row}) in (18) is:

$$NCW_{row} = FAR\_FDRI + (NCF_{CLB} + NCF_{DSP} + NCF_{BRAM} + 1) \times FR_{size} \tag{19}$$

where NCF_{CLB}, NCF_{DSP}, and NCF_{BRAM} are:

$$NCF_{CLB} = W_{CLB} \times CF_{CLB} \tag{20}$$

$$NCF_{DSP} = W_{DSP} \times CF_{DSP} \tag{21}$$

$$NCF_{BRAM} = W_{BRAM} \times CF_{BRAM} \tag{22}$$

The number of BRAM initialization words in a PRR row (NDW_{BRAM}) in (18) is:

$$NDW_{BRAM} = FAR\_FDRI + (W_{BRAM} \times DF_{BRAM} + 1) \times FR_{size} \tag{23}$$

Table VI. EXECUTION OF PLACE AND ROUTE OF PRMS USING THE AREA_GROUP ATTRIBUTE AND COMPARED AGAINST THE RESULTS IN TABLE V. THE PARENTHESIZED NUMBERS INDICATE THE RESOURCE SAVINGS/INCREASES WITH RESPECT TO TABLE V AS POSITIVE/NEGATIVE PERCENTAGES, RESPECTIVELY.

	Virtex-5 LX110T			**Virtex-6 LX75T**		
Parameter	**FIR**	**MIPS**	**SDRAM**	**FIR**	**MIPS**	**SDRAM**
LUT_FF_{req}	1082 (16.8%)	2183 (16.6%)	324 (2.4%)	999 (31.9%)	2630 (18.8%)	370 (3.9%)
DSP_{req}	32 (0%)	4 (0%)	0 (0%)	27 (0%)	4 (0%)	0 (0%)
$BRAM_{req}$	0 (0%)	6 (0%)	0 (0%)	0 (0%)	6 (0%)	0 (0%)
LUT_{req}	1015 (11.7%)	1528 (-0.1%)	191 (-21.7%)	999 (24.1%)	1932 (7.8%)	215 (-18.8%)
FF_{req}	410 (-4.1%)	1592 (0%)	292 (0%)	394 (0%)	1860 (0%)	324 (0%)
CLB_{req}	136 (16.6%)	273 (16.8%)	41 (2.4%)	125 (32.1%)	329 (18.8%)	47 (4.1%)
RU_{CLB}	68% (16.6%)	80% (16.8%)	68% (2.4%)	63% (32.1%)	75% (18.8%)	59% (4.1%)
RU_{FF}	26% (-4.1%)	59% (0%)	61% (0%)	12% (0%)	26% (0%)	25% (0%)
RU_{LUT}	63% (11.7%)	56% (-0.1%)	40% (-21.7%)	62% (24.1%)	55% (7.8%)	34% (-18.8%)
RU_{DSP}	80% (0%)	50% (0%)	0% (0%)	84% (0%)	25% (0%)	0% (0%)
RU_{BRAM}	0% (0%)	75% (0%)	0% (0%)	0% (0%)	75% (0%)	0% (0%)

IV. APPLICATION OF COST MODELS

We evaluate our cost models using two Xilinx devices (Virtex-5 LX110T and Virtex-6 LX75T) and three PRMs (FIR, MIPS, and SDRAM), formulas (1) to (17), and follow the flow in Fig. 1 to obtain the smallest PRR_{size} and the smallest partial bitstream size for each PRM. Our selected devices provide a broad generalization since these devices have disparate architectures (Table II and Table IV), where the Virtex-5 LX110T has 8 rows, the Virtex-6 LX75T has 3 rows, and both devices have different resource distributions. We note that since the Virtex-5 LX110T has only one DSP column in the device fabric, we use (4) instead of (3). To provide a good comparison, the selected PRMs have similar complexity and resource usage (with respect to CLBs, DSPs, and BRAMs) to the PRMs used in prior research [2][7]. FIR implements a finite impulse response (FIR) filter with 32 coefficients, MIPS implements a 5-stage pipeline of MIPS R2000 32-bit processor, and SDRAM implements a 32-bit synchronous dynamic random access memory (SDRAM) controller. The selected PRMs are not intended to work together as part of a complete PR design, but rather are selected to evaluate the accuracy of our cost models, which provides adequate verification and evaluation.

Results were obtained using a 1.8 GHz AMD Turion 64 Mobile ML-32, 2 GB RAM, and the Xilinx ISE 12.4 tools. Table V summarizes the results of our PRR size/organization cost model for the PRMs, where the values of H_{CLB}, W_{CLB}, H_{DSP}, W_{DSP}, H_{BRAM}, and W_{BRAM} for each PRM produce the smallest PRR_{size} and the highest RU for each resource.

In order to validate the accuracy of our PRR size/organization cost model, we specified area constraints (using the AREA_GROUP attribute in the user constraint file (*.ucf)) considering the position, size, and resource organization for an area on the target device (similar procedure as manual PRR floorplanning) where all associated PRM resources are used. Each PRM was considered as an entire design, and we used Xilinx ISE to place and route the PRM in the target device. All PRMs were fully placed and routed in the target devices, which means that the PRR size/organizations in Table V for each PRM were successfully placed and routed for the target devices.

We show the accuracy of our PRR size/organization cost model by comparing the results from Table V and the results from executing the entire design flow for the same PRMs and target devices. Table VI shows the results of executing Xilinx ISE to place and route the PRMs on the target devices from Table V using the AREA_GROUP attribute, and the comparison of these results against the results when using our PRR size/organization cost model (Table V). The numbers in parenthesis indicate resource savings/increases with respect to Table V as positive/negative percentages, respectively. The PRR sizes/organizations in Table VI are the same as in Table V. We note that the Xilinx tools perform optimizations to reduce the PRMs' resource requirements during place and route, resulting in fewer resources for the associated PRMs as compared to the resources included in the synthesis reports, and producing higher internal fragmentation (lower RU) in the PRRs, especially with LUTs and CLBs, but not with DSPs or BRAMs (0% change with respect to values in Table V).

To further reduce PRR size and increase RU, we further tested our PRR size/organization cost model with the LUT_FF_{req}, DSP_{req}, and $BRAM_{req}$ parameters from Table VI. LUT_FF_{req}, DSP_{req}, and $BRAM_{req}$ parameters are used in (1)-(7) to derive the PRR sizes/organizations for the associated PRMs. The PRR size/organization did not change for SDRAM for both device targets, we saved two/one CLB column(s) for the Virtex-5/Virtex-6 for FIR, respectively, and we saved two CLB columns in the Virtex-5 for MIPS, while MIPS failed place and route on the Virtex-6. These results show using the synthesis report values is sufficiently accurate for early estimation of the

Table VII. PARTIAL BITSTREAM SIZES (BYTES) FOR PRMS AND DEVICES FROM TABLE V

	Partial bitstream size (bytes)	
PRM	**Virtex-5 LX110T**	**Virtex-6 LX75T**
FIR	83440	77340
MIPS	157672	189140
SDRAM	18416	24204

Table VIII. EXECUTION TIMES IN MINUTES (m) AND SECONDS (s) FOR SYNTHESIS AND IMPLEMENTATION OF PRMS FOR THE PRR SIZE/ORGANIZATION FROM TABLE V

	Virtex-5 LX110T			**Virtex-6 LX75T**		
Process	**FIR**	**MIPS**	**SDRAM**	**FIR**	**MIPS**	**SDRAM**
Synthesis	4m 25s	4m 15s	3m 20s	4m	4m 50s	4m 23s
Implementation	5m 35s	5m 15s	2m 55s	4m 15s	5m 50s	4m 30s

PRR size and organization using our proposed cost model.

We note that high *RU*s lead to densely packed PRRs that may eventually cause routing problems in the PRR, but this depends on the complexity of the PRR's associated PRMs. Also, since the Xilinx tools allow the static region's nets to cross the PRRs, routing problems may arise if nets from the static region try to cross a densely packed PRR. Higher *RU*s may be obtained by selecting non-rectangular PRRs (such as an "L" or "T" PRR shape), but chances of routing problems in the PRRs are increased.

Using the values from Table IV for the PRMs and devices in Table V, and applying (18) to (23), we obtain the partial bitstream size in bytes for each PRM, which are depicted in Table VII. The obtained partial bitstream sizes are similar to those PRMs used in experiments to measure the reconfiguration times in prior work [2][4][7]. We note that there is no prior work that formulates a cost model to determine a partial bitstream size, or the most relevant prior work does not include detailed PRR resource usage information as we have included in our work, thus there is no direct comparison we can do with prior work.

Our cost models enable system designers to estimate the PRRs' sizes/organizations and associated PRMs' partial bitstream sizes without executing the entire PR design flow. Using our proposed methodology, system designers need only to synthesize the PRMs, and use the formulas from our proposed cost models. Table VIII depicts the execution times for synthesizing the PRMs from Table V on the selected devices, which includes the summation of the time to obtain the PRRs' sizes/organizations and bitstream sizes, which take less than 5 minutes in all cases. Table VIII also includes the implementation times of each PRM (using area constraints) for the PRR size/organizations in Table V as a reference.

V. Conclusions

In this work, we presented cost models for high-level, early-design-time estimation of partial reconfigurable region (PRR) size/organization and associated partial bitstream sizes for hardware-multitasking partially reconfigurable (PR) designs. Our high level approach precludes the typically lengthy PR design flow, and significantly aids system designers in PR partitioning and accelerates design space exploration. Our cost models are generally portable across different Xilinx field-programmable gate array (FPGA) families. Experimental results evaluated our cost models on two different Xilinx devices to determine the best PRR size/organization to produce the smallest PRR size and partial bitstream size for different PRMs. Our future work will use our cost models as part of the floorplanning stage in the PR design flow.

Acknowledgments

This work was supported by the Unidad Coordinadora del Programa de Ciencia y Tecnología (FINCyT), Perú, under contract N° 121-2009-FINCyT-BDE, by the I/UCRC Program of the National Science Foundation (NSF) under grants EEC-0642422 and IIP-1161022, and the NSF CHREC membership support of Draper Laboratory. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. The authors gratefully acknowledge the support of Universidad Nacional de Ingeniería – Lima, Perú, the Presidencia del Consejo de Ministros in Perú, through FINCyT, and the tools provided by Xilinx.

References

[1] C. Claus, B. Zhang, W. Stechele, L. Braun, M. Hübner, J. Becker, "A multi-platform controller allowing for maximum dynamic partial reconfiguration throughput," in *Proceedings of the International Conference on Field Programmable Logic an Applications (FPL'08)*, pp. 535-538, 2008.

[2] F. Duhem, F. Muller, P. Lorenzini, "Reconfiguration time overhead on field programmable gate arrays: reduction and cost model," *IET Computers & Digital Techniques*, vol. 6, no. 2, pp. 105–113, 2012.

[3] D. Koch, *Partial Reconfiguration on FPGAs: Architectures, Tools and Applications*, Lecture Notes in Electrical Engineering, vol. 153, Springer, 2013.

[4] M. Liu, W. Kuehn, Z. Lu, A. Jantsch, "Run-time partial reconfiguration speed investigation and architectural design space exploration," in *Proceedings of the 19th International Conference on Field Programmable Logic and Applications (FPL'09)*, pp. 498-502, 2009.

[5] A. Morales-Villanueva and A. Gordon-Ross, "On-chip context save and restore of hardware tasks on partially reconfigurable FPGAs," in *IEEE International Symposium on Field Programmable Custom Computing Machines (FCCM'13)*, pp. 61-64, 2013.

[6] A. Morales-Villanueva and A. Gordon-Ross, "HTR: on-chip hardware task relocation for partially reconfigurable FPGAs," in *Proceedings of the 9th International Symposium on Applied Reconfigurable Computing (ARC'13)*, LNCS, vol. 7806, pp. 185-196, 2013.

[7] K. Papadimitriou, A. Dollas, S. Hauck, "Performance of partial reconfiguration in FPGA systems: a survey and cost model," *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, vol. 4, no. 4, article 36, 2011.

[8] Xilinx Inc., Partial Reconfiguration of Virtex FPGAs in ISE 12 (WP374), July 23, 2010.

[9] Xilinx Inc., Virtex-5 FPGA Configuration User Guide v3.10 (UG191), Nov 18, 2011.

[10] Xilinx Inc., Virtex-5 FPGA User Guide v5.4 (UG190), March 16, 2012.