Bit density-based pre-characterization of RAM cells for area critical SOC design

Dilip Kumar Ajay (dajay@qti.qualcomm.com)

accellera SYSTEMS INITIATIVE

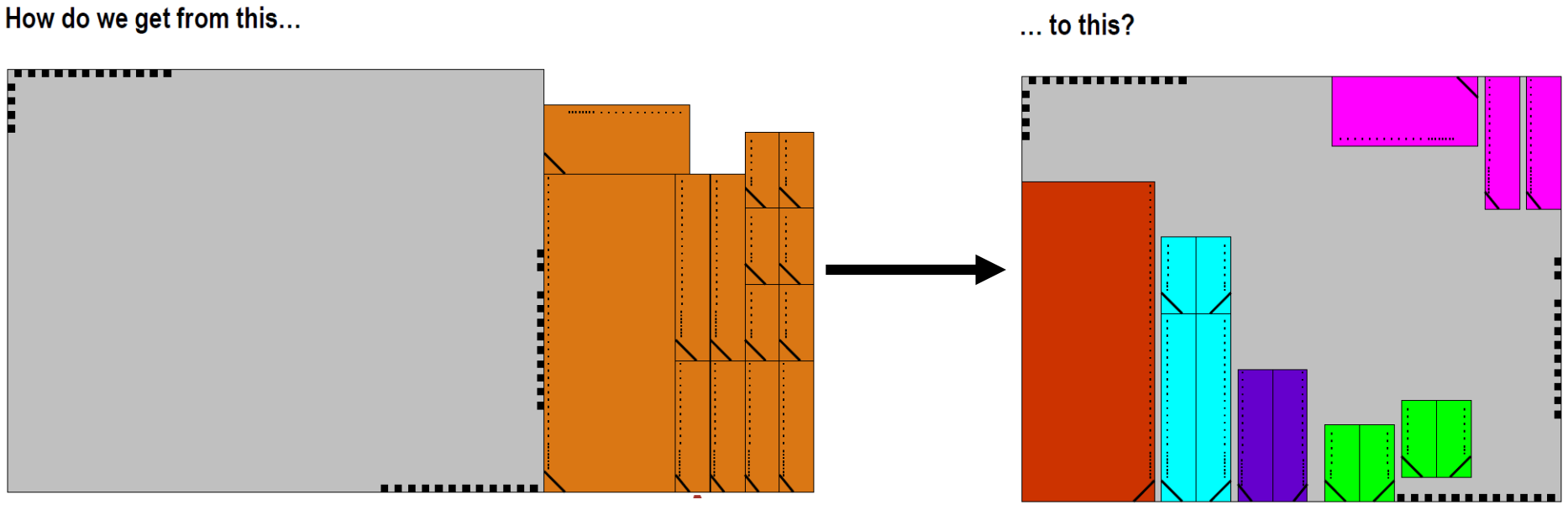
Qualcomm

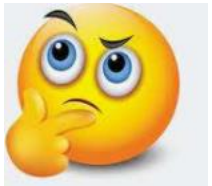
2020 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION EUROPE

Agenda

- Problem statement
- Invention to address the problem
- Test case
- Conclusion

Traditional approach





- But the larger question should be, is the memory configuration chosen for this design the best?

What are we solving?

- Memories constitutes a significant percentage of almost all SOC's die area. For a given logical depth, logical width and cycle time requirement, we get many memory configurations with different values for access time, physical width, physical depth and area.

1	compiler	words	bits	mux	mvt	metal_	flexible	Cyclet	Access	geomx	geomy	area
5	sram_sp_uhde_svt_mvt	16384	74	8	HS	2Xa1Xd	8	1.021	0.987	461.835000	262.320000	121148.557
6	sram_sp_uhde_svt_mvt	16384	74	8	LP	2Xa1Xd	8	1.198	1.176	461.835000	262.320000	121148.557
7	sram_sp_uhde_svt_mvt	16384	74	16	HS	2Xa1Xd	4	1.014	0.912	241.785000	524.352000	126780.448
8	sram_sp_uhde_svt_mvt	16384	74	16	LP	2Xa1Xd	4	1.162	1.127	241.785000	524.352000	126780.448
9	sram_sp_hde_svt_mvt	16384	74	8	HS	2Xa1Xd	8	0.793	0.833	430.065000	318.480000	136967.101
10	sram_sp_hde_svt_mvt	16384	74	8	LP	2Xa1Xd	8	0.915	0.991	430.065000	318.480000	136967.101

- Today the decision to pick the right RAM cell for a given design is iterative and manual. Existing techniques for this mostly involve designer feedback which is a slow, reactive and incremental approach.

Agenda

- Problem statement
- Invention to address the problem
- Test case
- Conclusion

Key idea

- score each library cells based on area characteristics. This is a one-time processing effort before the synthesis flow.
- Multiple scores are possible per library cell for the range of expected values for different parameters available during memory compilation.
- The library cell score for area optimization is then used in the synthesis flow. The pre-characterization scores are used to estimate/optimize the area profile.

Scoring scheme for RAM cells

- Area Difficulty Score (ADS): is a parameter or a weight given to each RAM cell of the library based on how much each RAM cell is likely to contribute towards the area. Area difficulty score of each RAM cell is a function of number of words, number of bits, physical width and depth.

ADS of a RAM cell = F (words, bits, geom_x, geom_y) (1)

- ADS of a RAM cell is directly proportional to a new parameter we are introducing called "Megabits per mm square", which is nothing but bit density which will determine the area efficiency of each RAM cell.

Calculating Megabits per mm square (Mbits/mm^2)

Area of RAM cell in mm^2 = [(geom_x) * (geom_y)]/1000000 (2)

Megabits = [(words) * (bits)]/1048576 (3)

Mbits/mm^2 =[(Eq.3) / (Eq.2)] (4)

ADS α [1/A_t], where A_t, is the access time of RAM cell. (5)

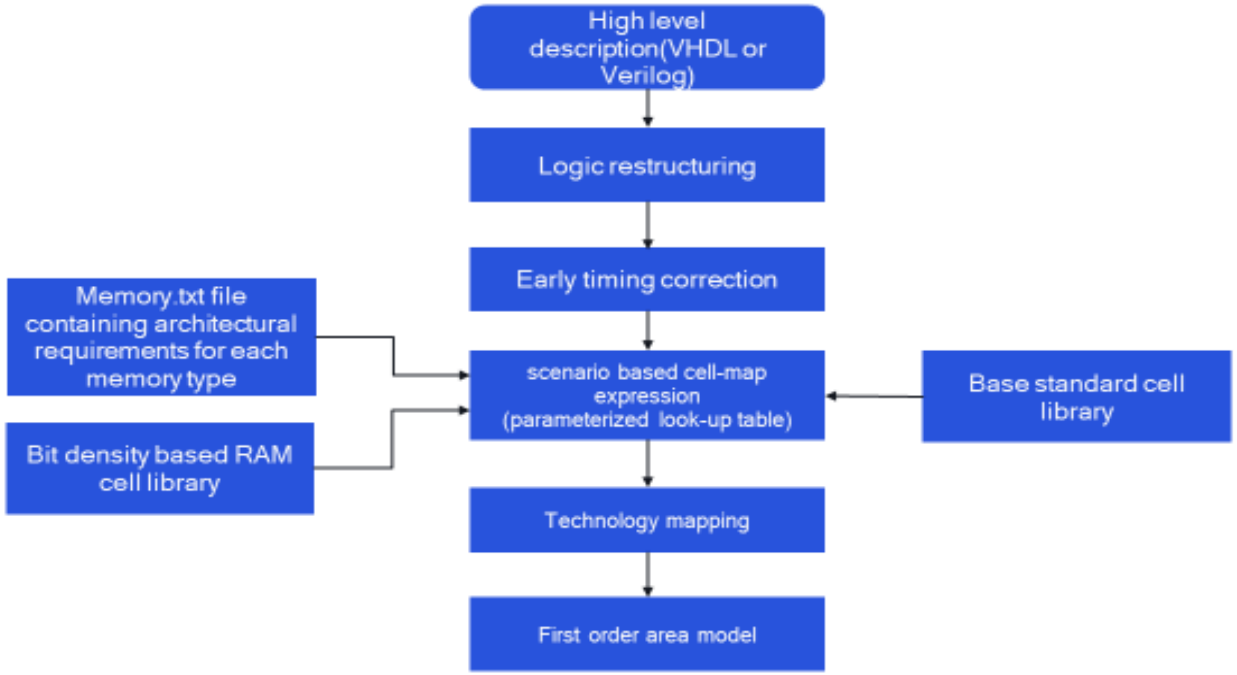
Cycle time C_t of the RAM cell is treated as constant.

Therefore, we can say area difficulty score of each RAM cell can be derived from

ADS = [(C_t * Mbits/mm^2) / A_t] (6)

Higher the Mbits/mm2 value, better it is for area optimization. Hence, we can now pre-characterize or score RAM cells based on Mbits/mm2 value.

Bit density aware synthesis flow



There are 2 additional steps which are introduced to the traditional design flow: provide bit-density aware pre-characterized RAM cell library and provide a memory.txt file with logic requirement for the memory used in the module.

Agenda

- Problem statement
- Invention to address the problem
- Test case
- Conclusion

Example

Let’s take a case where the logical requirement for the memory used in the module is logic depth of 21504 and logic width of 266. In each write cycle we need to write 160B of data into 3 banks of memory. The aspect ratio of the block is x=5000 micron and y=1650 micron.

Pre-characterized RAM cell library for given example

compiler	words	bits	mux	mvt	metal_	flexible	Cyclet	Acces	geomx	geomy	area	Mbits	area in mm	Mbits/mmSqua
sram_sp_uhde_svt_mvt	16384	74	8	HS	2Xa1Xd	8	1.021	0.987	461.835000	262.320000	121148.557	1.15625000	0.121148557	9.544067454
sram_sp_uhde_svt_mvt	16384	74	8	LP	2Xa1Xd	8	1.198	1.176	461.835000	262.320000	121148.557	1.15625000	0.121148557	9.544067454
sram_sp_uhde_svt_mvt	16384	74	16	HS	2Xa1Xd	4	1.014	0.912	241.785000	524.352000	126780.448	1.15625000	0.126780448	9.12009713
sram_sp_uhde_svt_mvt	16384	74	16	LP	2Xa1Xd	4	1.162	1.127	241.785000	524.352000	126780.448	1.15625000	0.126780448	9.12009713
sram_sp_hde_svt_mvt	16384	74	8	HS	2Xa1Xd	8	0.793	0.833	430.065000	318.480000	136967.101	1.15625000	0.136967101	8.441808227
sram_sp_hde_svt_mvt	16384	74	8	LP	2Xa1Xd	8	0.915	0.991	430.065000	318.480000	136967.101	1.15625000	0.136967101	8.441808227

- We can see from the above table the number of options available for synthesis tool for the given memory specification. Since the synthesis tool is physically aware, it can read the size of the module, which is 5000x1650 microns.
- It is evident from the above table, that the synthesis tool can easily pick the memory configuration with highest Mbits/mm^2, which would otherwise be user input based on floorplanning results and can be iterative.

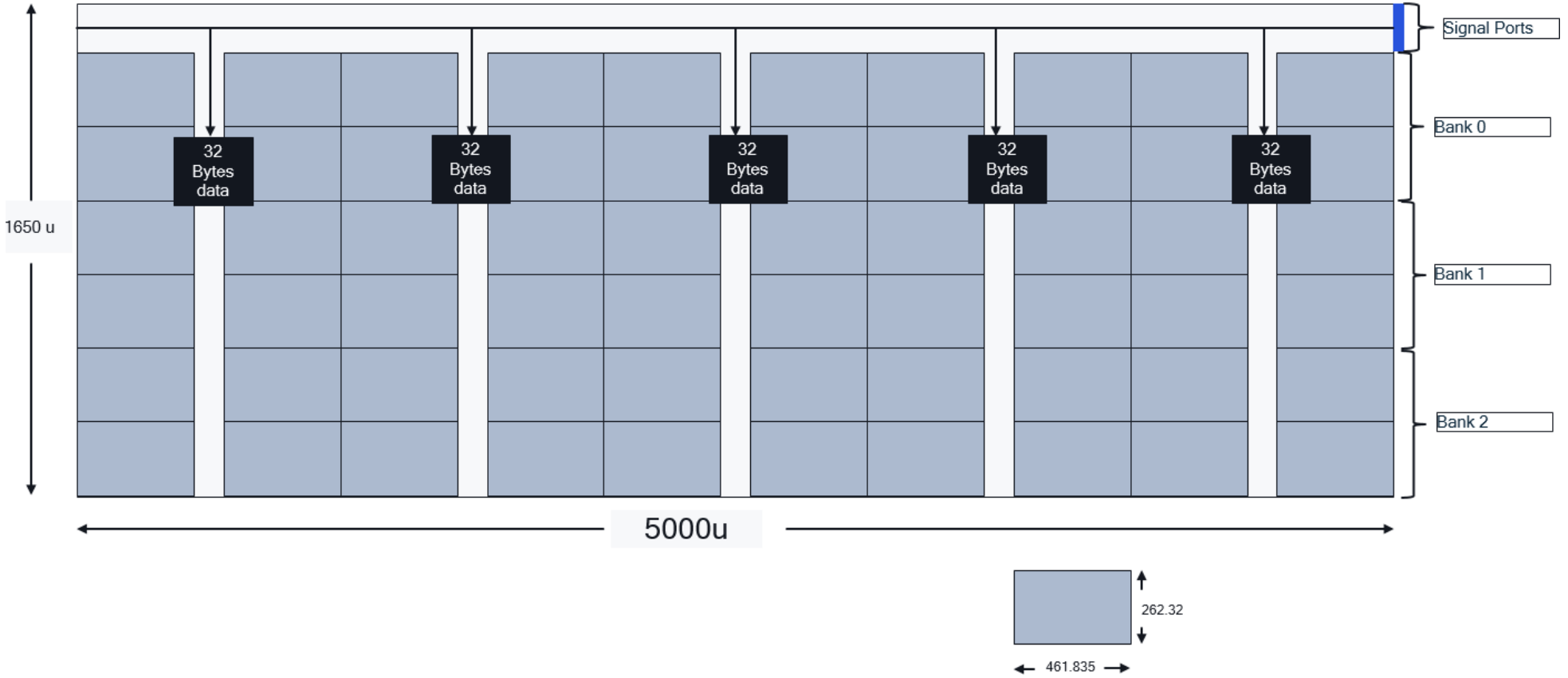
Content of memory.txt for given example

```
#-----------------------------------------------------------------#
# Description:
# vendor=VENDOR type=TYPE vt=VT depth=DEPTH width=WIDTH mux_factor=MF write_mask=WM repaircols=RCOLS pipeline=PL \
# [banks=BANKS] wrapper_name=WNAME logical_width=LWIDTH width_inst=WINST

# Memory related options:
# VENDOR=[arm]
# TYPE=[sr1p_hd|sr2p_hd|sr2p_uhd|rf1p_hd|rf2p_hs]
# VT=[svt|hvt]      default: hvt
# MF= [2|4|8|16|32]
# WM=[yes|no] default: no
# PL=[yes|no] default: no
# BANKS= [1|2|4|8] - only applicable to sr2p_uhd memory

# Memory and wrapper related options:
# RCOLS= [0|1|2|3]
# Wrapper related options:
# WNAME=verilog module name
# LWIDTH=[integer]
# WINST=[integer]
################ PD XYZ mems ###############
vendor=arm type=sr1p_uhd vt=hs depth=16384  width=74 banks=8 mux_factor=8  write_mask=no repaircols=2 pipeline=ext logical_width=266 logical_depth=21504
wrapper_name=XYZ
```

Floorplan inputs

- Since we need to write 160B (1280 bits) of data in each cycle, into 3 banks, we can see that we would need 4 instances of each RAM cell selected to realize logic depth of 21k and logic width of 266 (256 bits of data + 10 bits of ECC logic) plus 2 repair columns.
- Therefore we would need 5*4 instances to realize a logic width of 1280 bits or 160 Bytes. Since we need to write 160 Bytes of data in 3 banks, the total instances needed for this module to meet the aspect ratio and logical requirement would be 60 instances.

Final floorplan



Agenda

- Problem statement
- Invention to address the problem
- Test case
- Conclusion

Summary

- A new approach is proposed which can easily be integrated into existing physical aware synthesis tool.
- The pre-characterized RAM cell based on bit-density aids synthesis tool during technology mapping step to intelligently select the best RAM cell for the given module for the aspect ratio and logical requirements, which would otherwise be iterative and error prone saving a lot of turnaround time

Questions?