Optimization of System-on-Chip Platform using modeFRONTIER

Sara Bocchio,
AST – Ultra Low Power Platform
STMicroelectronics
Agrate Brianza (MI) Italy
sara.bocchio@st.com

Outline

- SoC introduction
- MULTICUBE experience
  - Integration and exploration with ModeFrontier
  - Results
- Concluding remarks
- The next challenge…

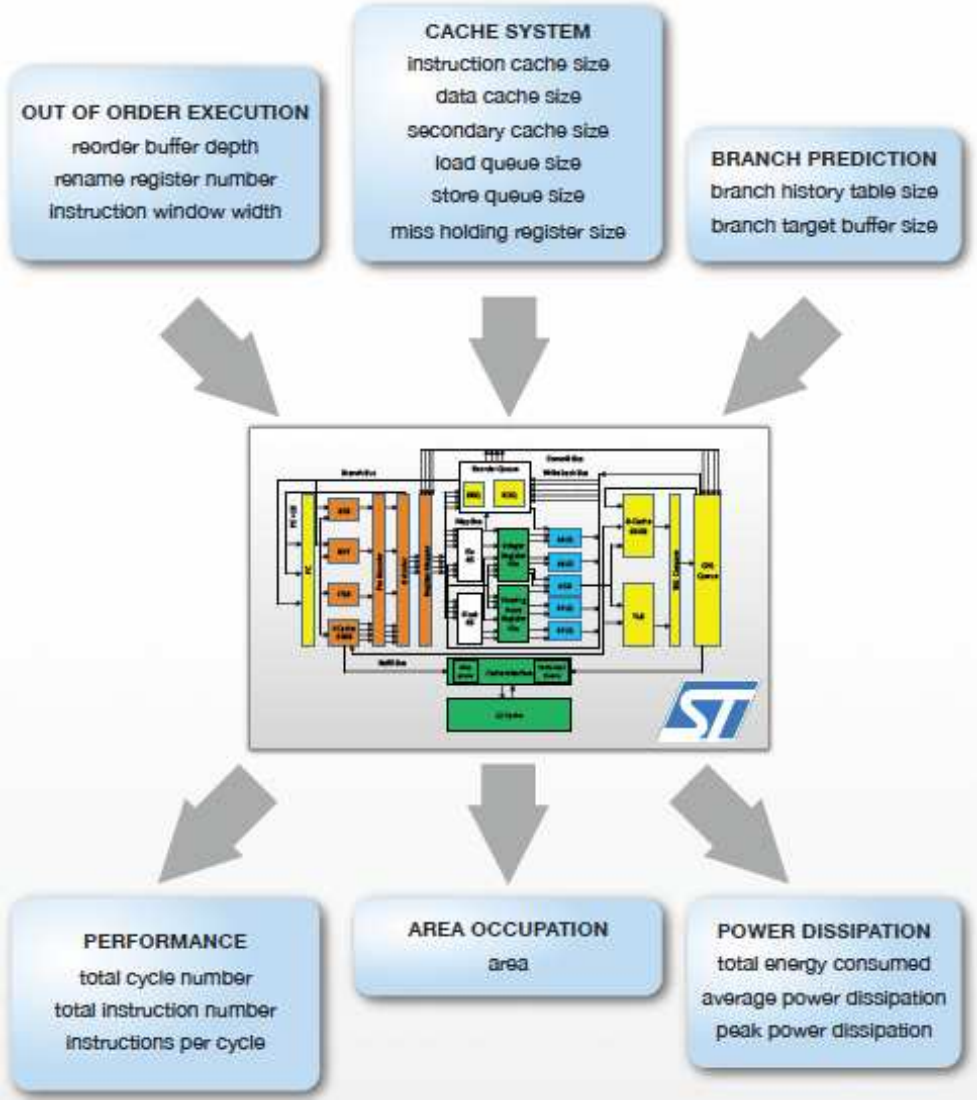
System on Chip

- System on Chip (SoC) is the integration of all components of a electronic system into a single integrated circuit (chip).
- It may contain
  - digital,
  - analog,
  - mixed-signal,
  - radio-frequency functions
- Target of SoC design is a **performance/power/area tradeoff**

MULTICUBE project

- FP7 project End June 2010
- Focused on multi-objective Design Space Exploration (DSE) framework for System-on-Chip architecture
  - Bring discrete domain into modeFRONTIER

Low power processor- design space



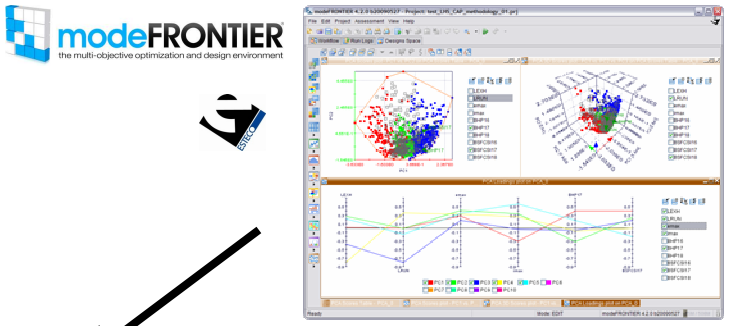
- Design Space
  - out-of-order engine
    - Reorder buffer depth (32, 48, 64, 80, 96, 112 or 128)
    - Rename register number (
      48, 64)
    - Instruction window
      24, 32)
  - cache system
    - Cache si
      16KB,
      256
      nd D-cache:
      ache: 0,
      KB).
    - e size (16, 24 and
      ss holding register) size
  - n prediction unit
    - BHT (branch history table) size (512, 1024, 2048 and 4096)
    - BTB (branch target buffer) size (16, 32, 64 and 128)

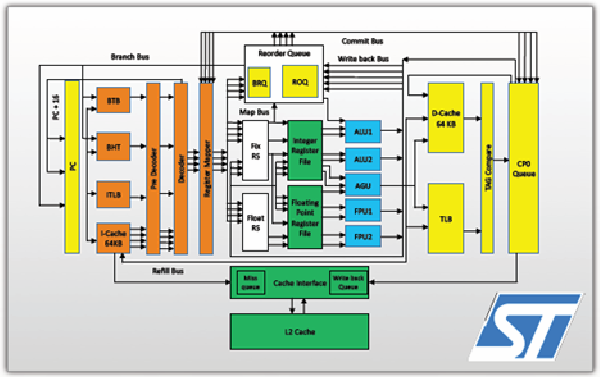
Total = 1,161,216 design points

Integration with ModeFRONTIER

A **design space exploration** analyzing a set of random selected designs was performed with **M3Explorer** for statistical analysis.

All designs in the reduced design space were evaluated by performing a full factorial multi-level exploration with **modeFRONTIER** on a computer cluster at ST, obtaining the **real Pareto front**. The full factorial was performed on 5000 points - 2,5 day simulations on ST computer farm





First results and some considerations

Parameter	Type	Minimum	Maximum	power_dissipation	Cpi
rob_depth	scalar	32	128	-0.44%	-0.28%
rmreg_cnt	scalar	16	64	-0.42%	-3.96%
iw_width	scalar	8	32	0.06%	-5.52%
icache_size	scalar	16	64	2.11%	-4.80%
dcache_size	scalar	16	64	6.76%	-7.90%
scache_size	scalar	0	1024	1.02%	0.54%
lq_size	scalar	16	32	-0.23%	1.08%
sq_size	scalar	16	32	0.09%	-0.14%
mshr_size	scalar	4	8	-0.16%	-0.20%
bht_size	scalar	512	4096	-0.13%	-9.35%
btb_size	scalar	16	128	0.22%	-5.56%

Design of Experiments

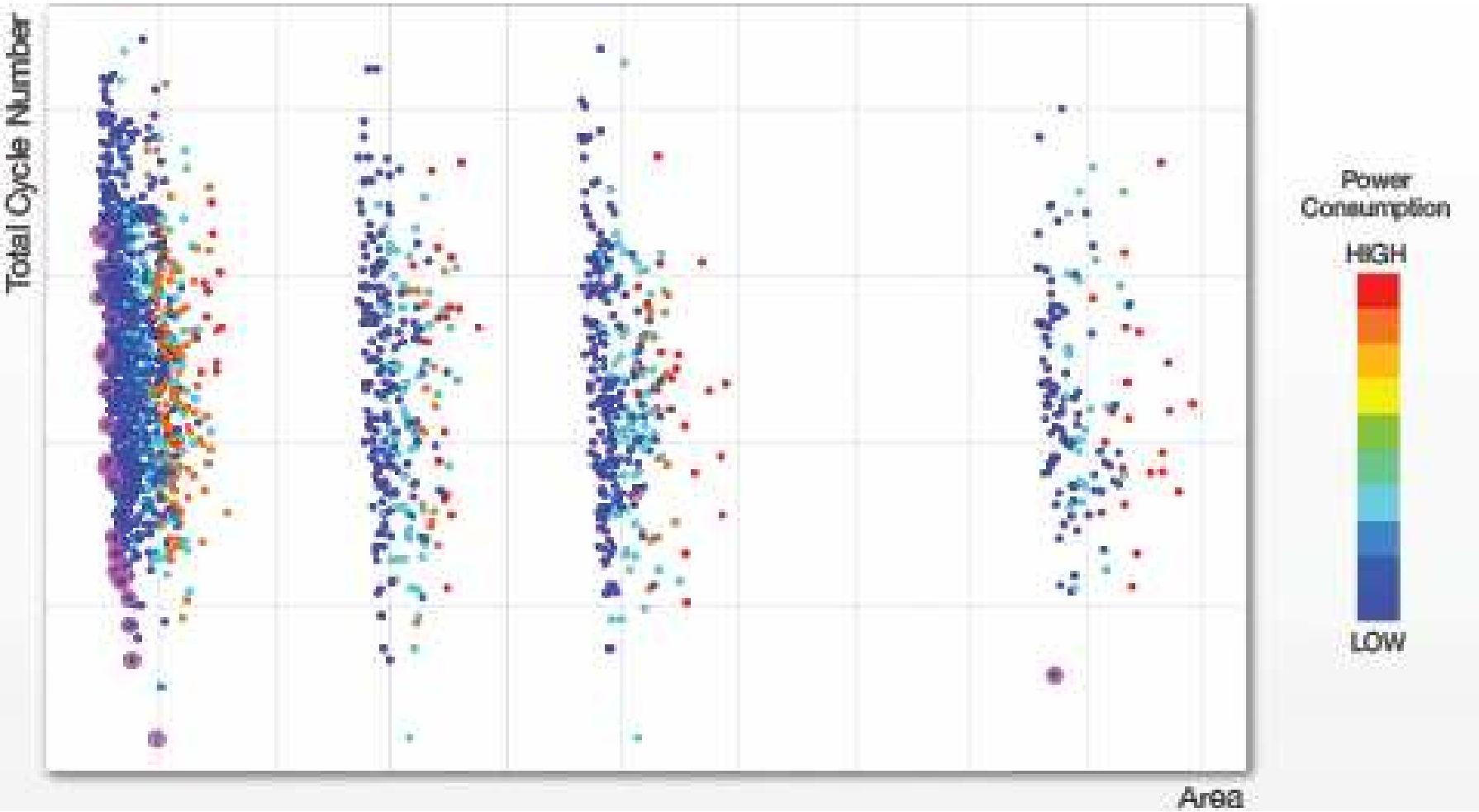
- Dimensionality reduction:

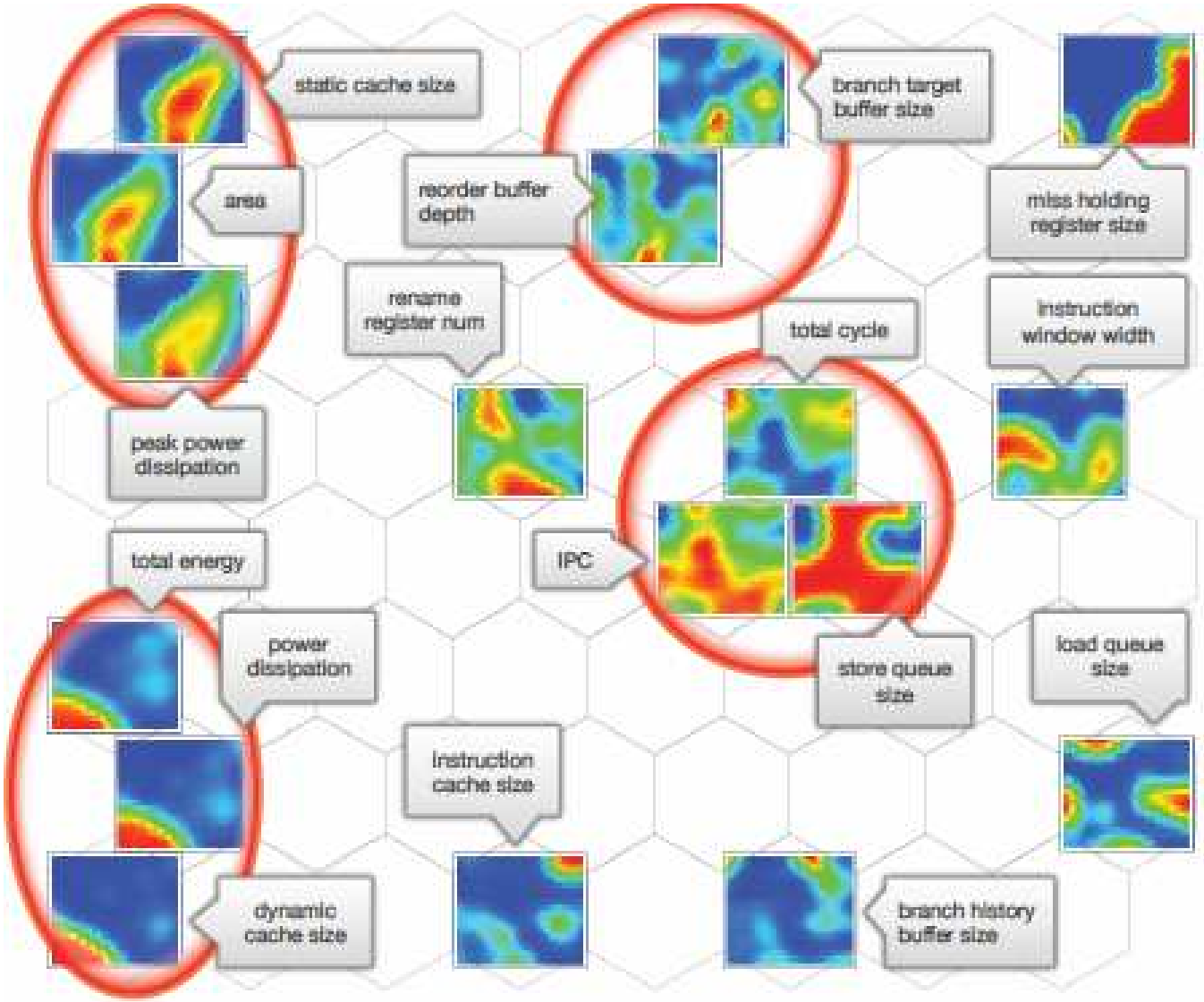
Input Variable

	Name	Variable T...	Value	Expression	Distribution	Scale	Lower Bo...	Upper Bo...	Central Va...	Delta Value	Base	Step	Tolerance	Format	Arrangem...
0	k rob_de...	Constant	32.0		None	0	32.0	128.0	80.0	48.0	7	16.0	0.0	#0	Ordered
1	rmreg_cnt	Variable	0.0		None	0	16.0	64.0	40.0	24.0	4	16.0	0.0	#0	Ordered
2	iw_width	Variable	0.0		None	0	0.0	3.0	1.5	1.5	4	1.0	0.0	#0	Unordered
3	icache_si...	Variable	0.0		None	0	4.0	6.0	5.0	1.0	3	1.0	0.0	#0	Ordered
4	dcache_s...	Variable	0.0		None	0	4.0	6.0	5.0	1.0	3	1.0	0.0	#0	Ordered
5	scache_s...	Variable	0.0		None	0	0.0	3.0	1.5	1.5	4	1.0	0.0	#0	Unordered
6	k lq_size	Constant	16.0		None	0	16.0	32.0	24.0	8.0	3	8.0	0.0	#0	Ordered
7	k sq_size	Constant	16.0		None	0	16.0	32.0	24.0	8.0	3	8.0	0.0	#0	Ordered
8	k mshr_s...	Constant	2.0		None	0	2.0	3.0	2.5	0.5	2	1.0	0.0	#0	Ordered
9	bht_size_...	Variable	0.0		None	0	9.0	12.0	10.5	1.5	4	1.0	0.0	#0	Ordered
10	btb_size_...	Variable	0.0		None	0	4.0	7.0	5.5	1.5	4	1.0	0.0	#0	Ordered

- Doe = 9216 points => now a complete DSE can be performed!
- Not all the 9216 should be simulated since we had already results from previous set

Final results




static cache size
area
peak power dissipation
branch target buffer size
reorder buffer depth
miss holding register size
rename register num
total cycle
instruction window width
total energy
IPC
power dissipation
store queue size
load queue size
instruction cache size
dynamic cache size
branch history buffer size

Concluding remarks

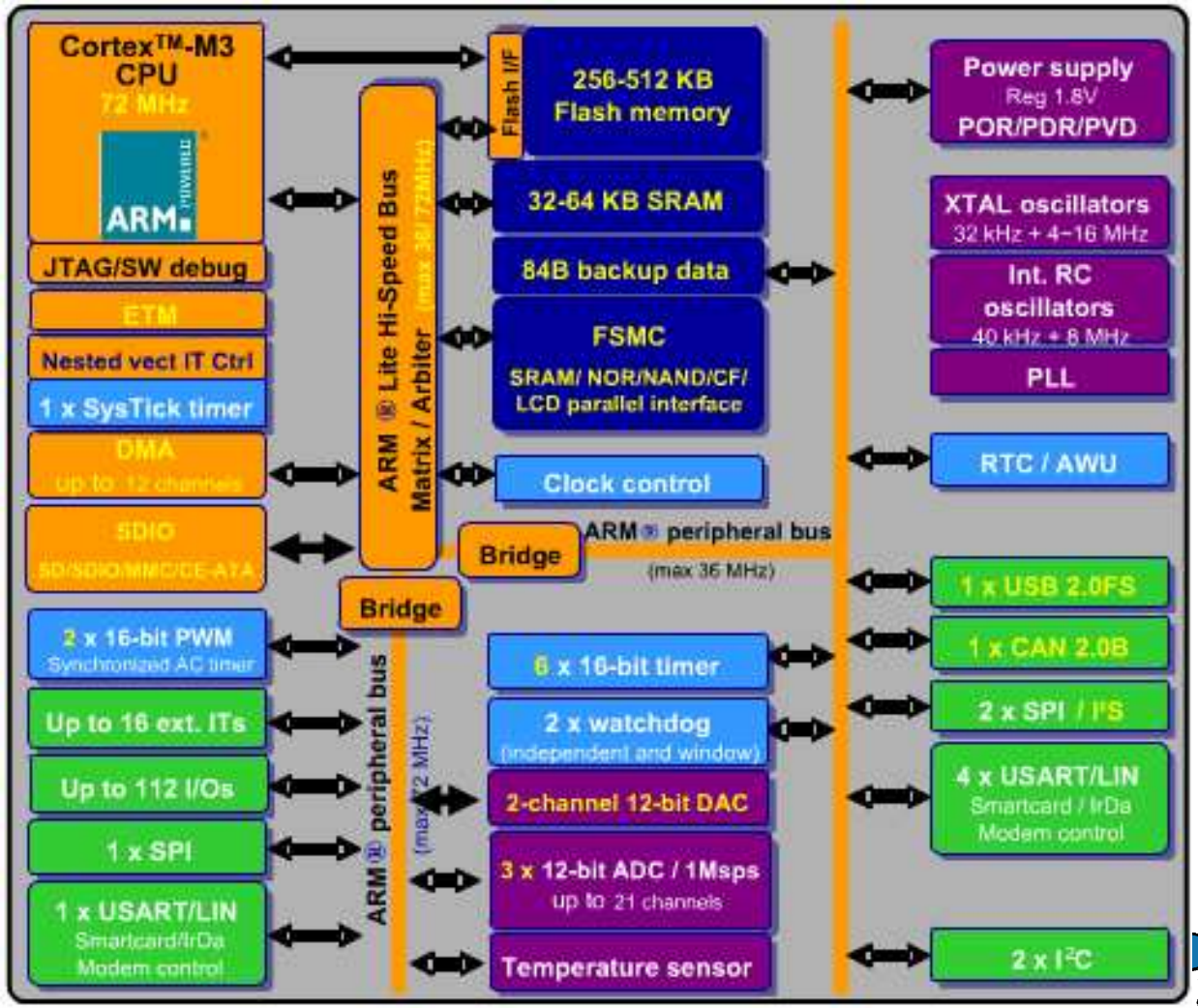
- A first coarse grain DSE can reduce DoE points
  - Designer experience improve confidence about the removal of the parameter
- Optimization algorithm can also help reduce DoE
- Reduce DoE means reduce time-to-market

The end?

- This is a “simple” example
  - Optimization of a single SoC IP (the core)
  - Objective is a single point (“the solution”)
- What can be a real scenario? (i.e. product line)

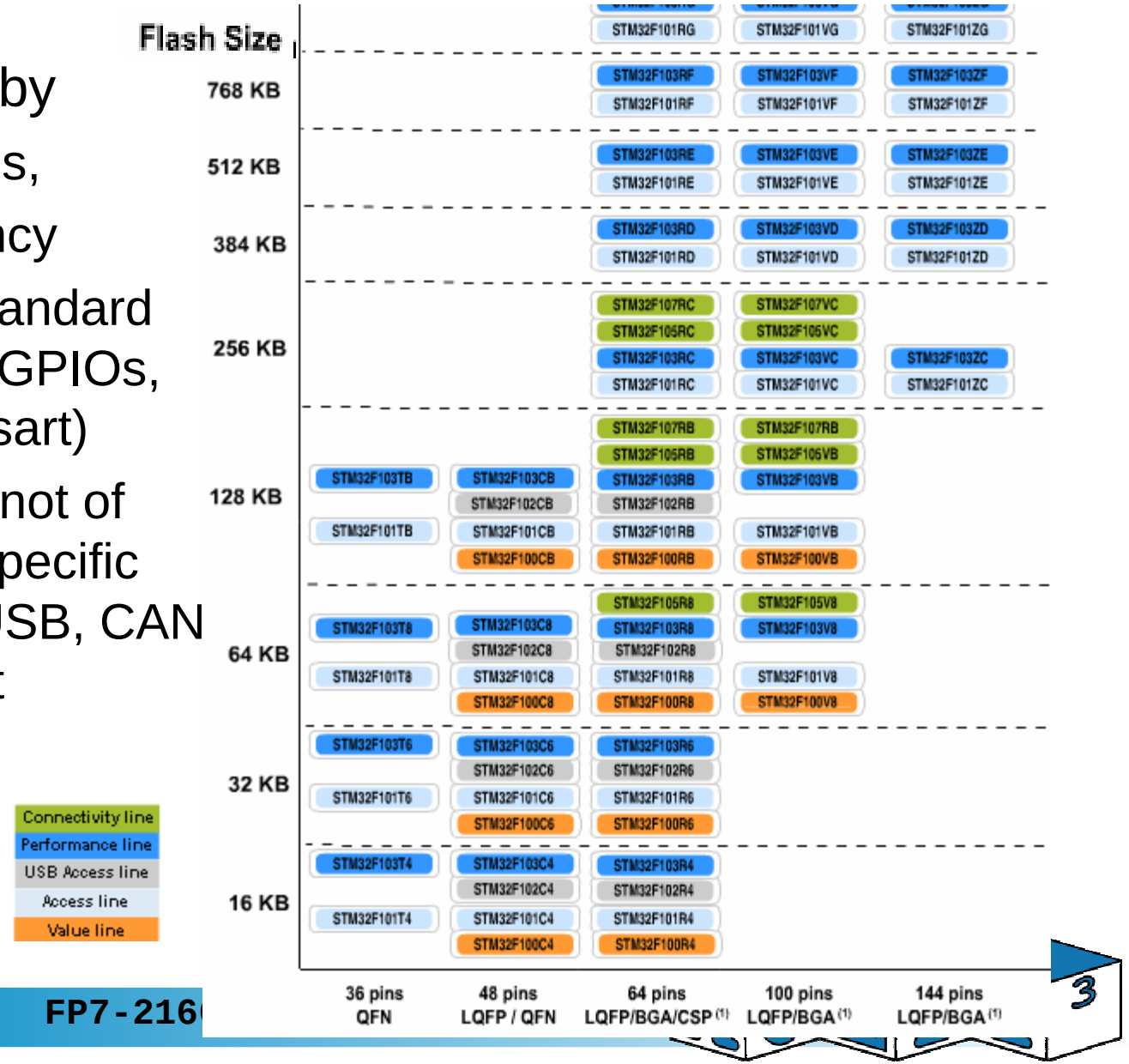
STM32 Microcontrollers

http://www.st.com/mcu/inchtml-pages-stm32.html



And the product portfolio

- Differentiation by
  - memory sizes,
  - CPU frequency
  - number of standard peripherals (GPIOs, timers spi, usart)
  - inclusion or not of application specific peripheral, USB, CAN and Ethernet



What is the best solution?