

PHILIPS

Vector Processing as an Enabler for Software Defined Radio in Handheld Devices

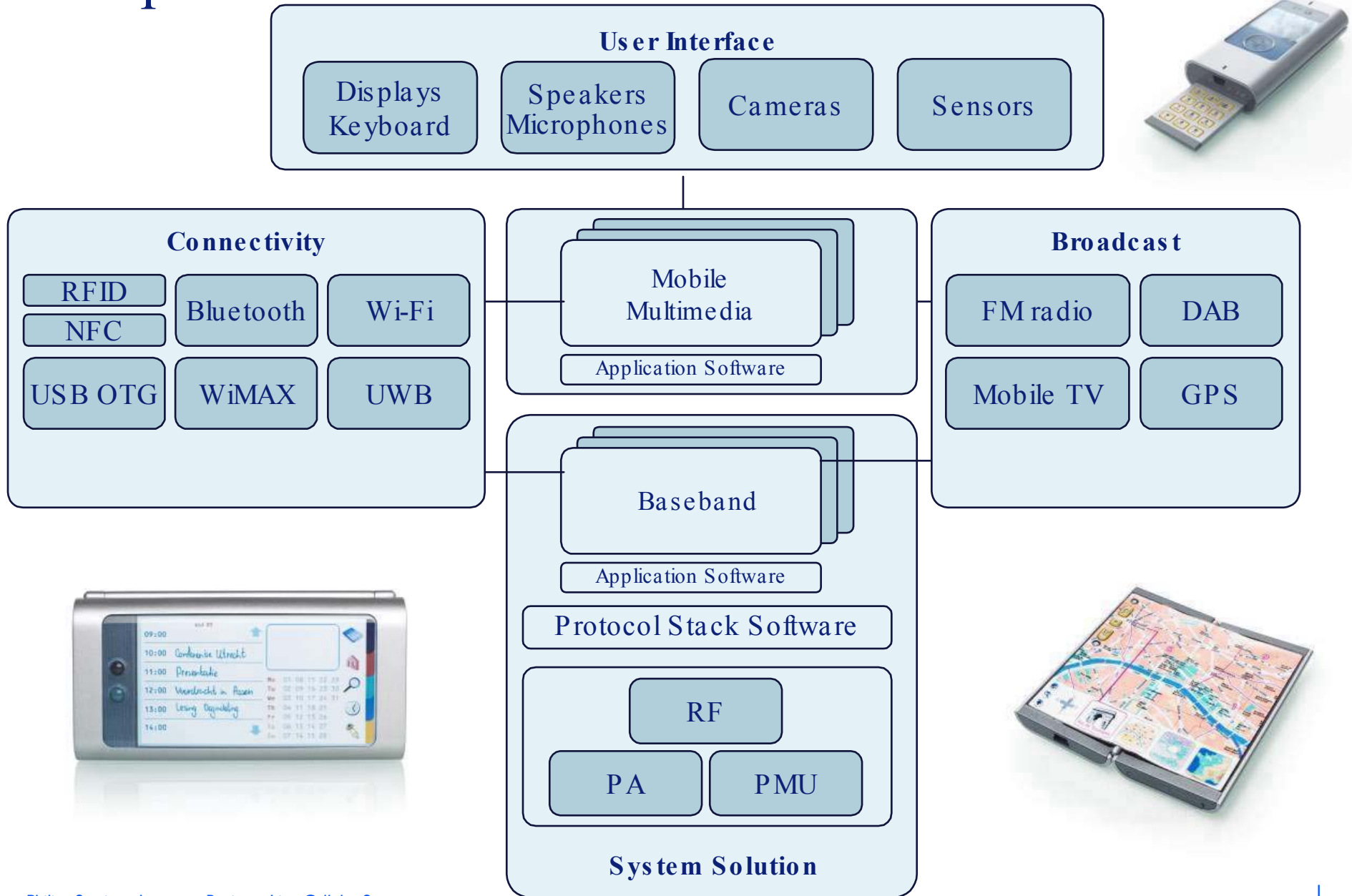
Kees van Berkel^{1,2}

Frank Heinle³, Patrick Meuwissen¹, Kees Moerman³, Matthias Weiss³

















¹ Philips Research Laboratories Eindhoven, ² Technical University Eindhoven

³ Philips Semiconductors

Nexperia Mobile



Full set of new products with Philips inside

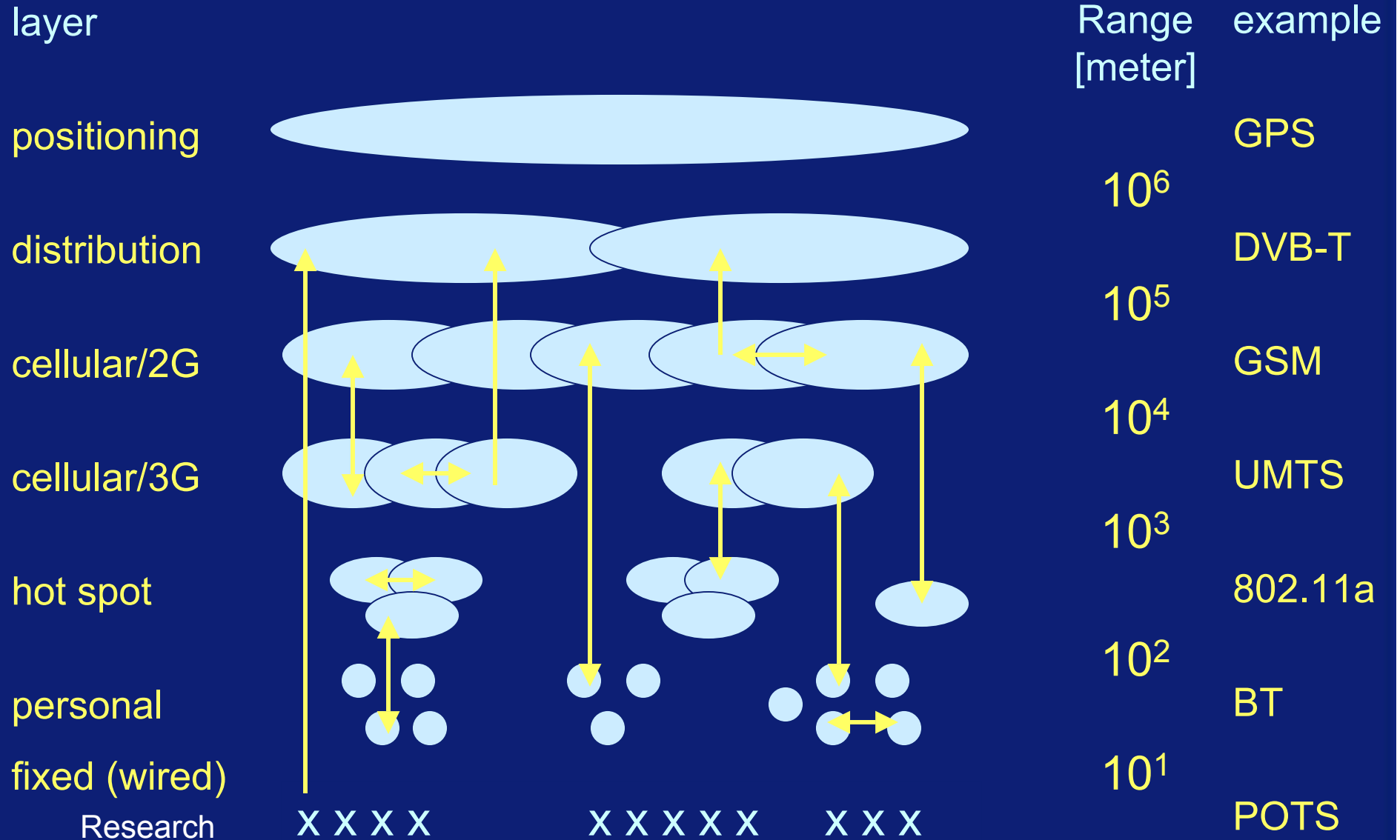
 	 	 		 
<p>Philips Semiconductors 2004 Market Share</p> <ul style="list-style-type: none"> • System Solution: > 1 out of 10 GSM/GPRS • BB: 1 out of 7 GSM/GPRS • RF: 1 out of 6 GSM/GPRS • PMU: 1 out of 5 GSM/GPRS • RF 3G: 1 out of 3 3G-Phones 				
 		 <p><small>Arima Computer Corporation</small></p>		 



Contents

- **Software-Defined Radio (baseband part):**
 - background, motivation
 - HW architecture, analysis of flexibility needs
- **Vector processing as key enabling technology:**
 - requirements
 - architectures: (OnDSP), EVP
 - results, benchmarking
- **Application of vector processing to SDR/bb**
- **Software Defined Radio, virtualization ...**

A layered, seamless network





4G = “always best connected”

(ABC, Ericsson; “best” from whose perspective?)

= best combination of links & link parameters (e.g. bit rate)
{GSM, UMTS, WLAN, UWB, WIMAX, DVB-H, BT, ...}
– including seamless handover

where “best” is based on:

- personal preferences (e.g. quality vs costs),
- user profile (e.g. known to operator),
- terminal capabilities (e.g. screen size, energy),
- requirements from active applications,
- location of user,
- network characteristics (bit rate, costs, QoS)
- channel conditions, etc.



A layered, seamless network: many standards!

- Positioning GPS, Galileo
- Distribution (broadcast) DAB, DVB-T, DVB-H, DMB-T, ISDB-T
- Cellular 2G GSM, IS-95, IS-136, PHS, EDGE, GPRS, IS-136+
- Cellular 3G UMTS, CDMA2000, TD-SCDMA, HSDPA
- Hot spot 802.11 a, b, g, n, (802.16a, e)
- Personal Area Network DECT, BlueTooth, UWB
- Fixed (wired) POTS, USB, firewire



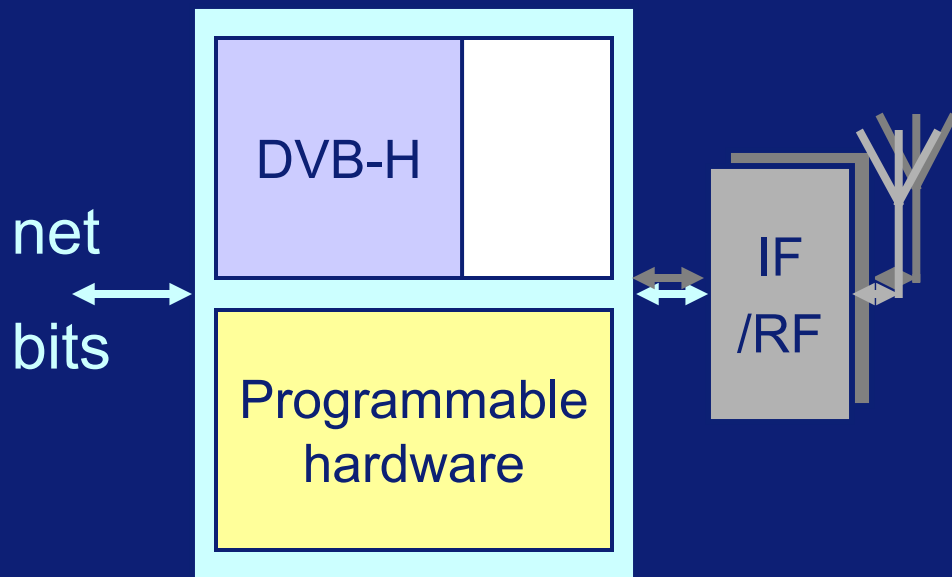
Many uncertainties

- which standards will be viable/winning?
- which combination(s) of standards will end up in a single device? When? Market volumes?
- which standards can be simultaneously active in a single device?
- how will these standards evolve?

Very hard to predict!

Hence, a generic (flexible) architecture is needed.

Software-Defined Radio (SDR)



SDR supports:

- multiple standards,
- tracking of standard evolution and algorithm improvement,
- multi-mode operation,
- upgrading in the field;

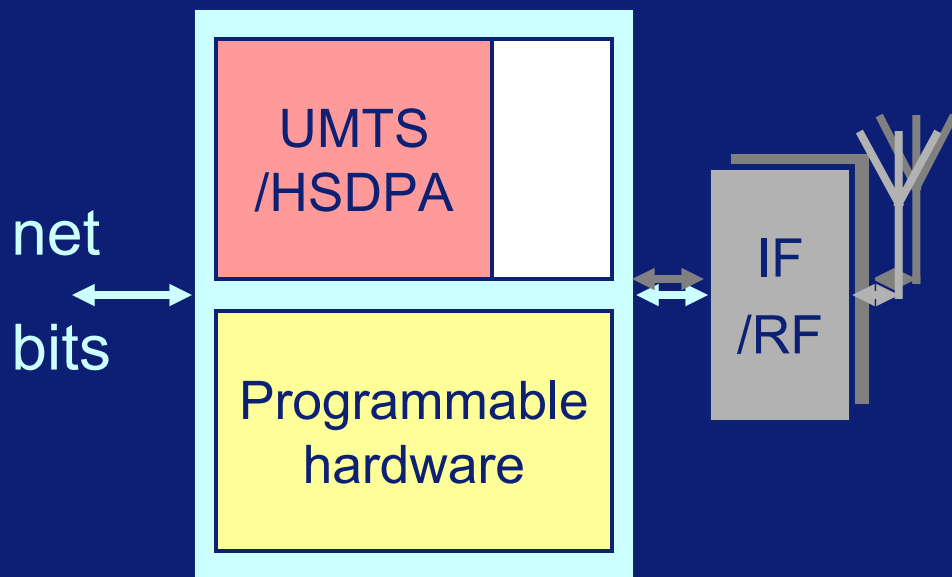
and must be:

- competitive in costs versus a dedicated hardware modem,
- low power (active + standby),
- conveniently programmable.

www.sdrforum.org

Research

Software-Defined Radio (SDR)



SDR supports:

- multiple standards,
- tracking of standard evolution and algorithm improvement,
- multi-mode operation,
- upgrading in the field;

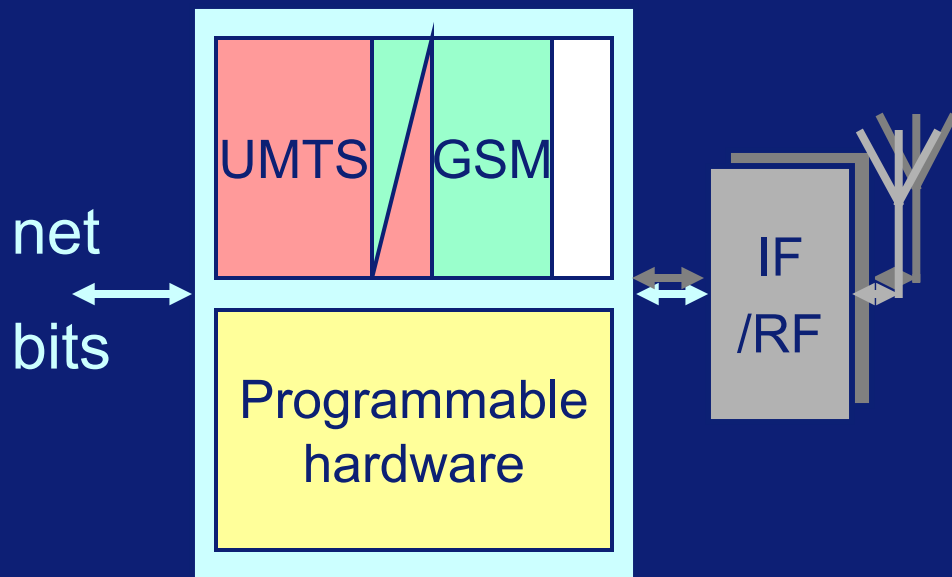
and must be:

- competitive in costs versus a dedicated hardware modem,
- low power (active + standby),
- conveniently programmable.

www.sdrforum.org

Research

Software-Defined Radio (SDR)



SDR supports:

- multiple standards,
- tracking of standard evolution and algorithm improvement,
- multi-mode operation,
- upgrading in the field;

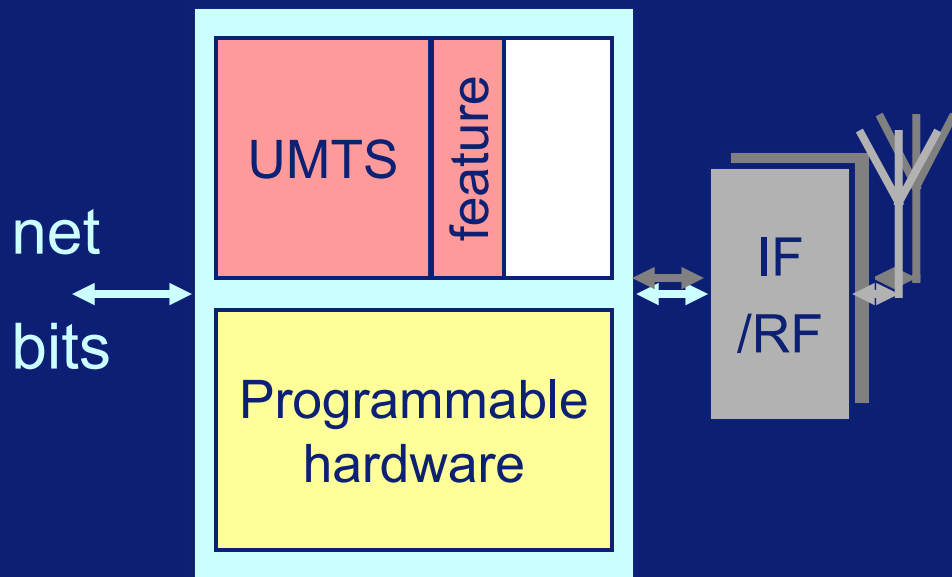
and must be:

- competitive in costs versus a dedicated hardware modem,
- low power (active + standby),
- conveniently programmable.

www.sdrforum.org

Research

Software-Defined Radio (SDR)



SDR supports:

- multiple standards,
- tracking of standard evolution and algorithm improvement,
- multi-mode operation,
- upgrading in the field;

and must be:

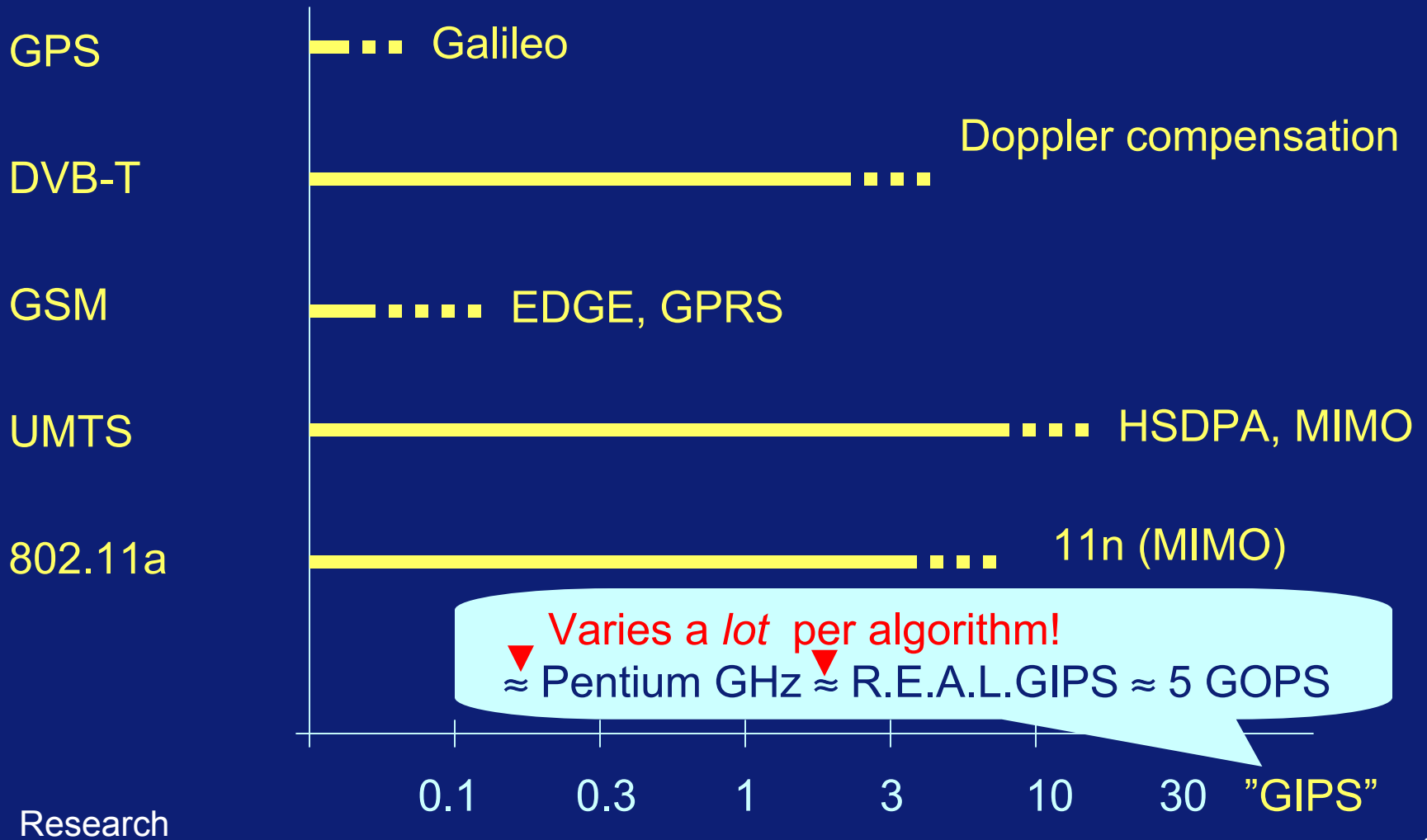
- competitive in costs versus a dedicated hardware modem,
- low power (active + standby),
- conveniently programmable.

www.sdrforum.org

Research



SDR: computational load



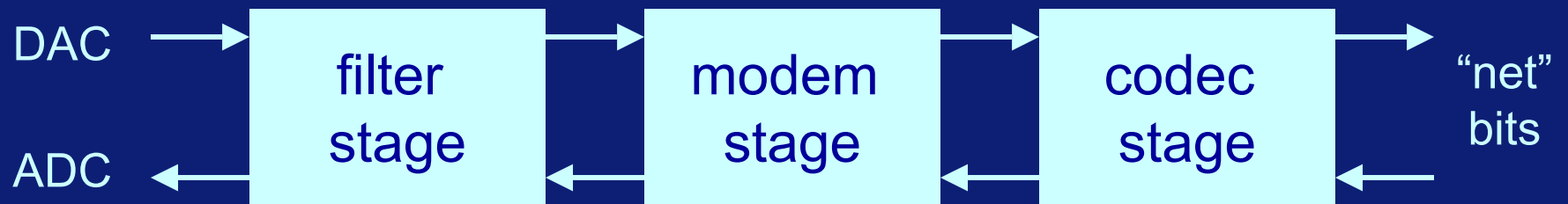


SDR: computational load for UMTS

- 10+ GIPS !
- 10 × Mobile Pentium @ 1 GHz?
 - still requires hand-optimized C code
 - silicon area: 100 times too much
 - > 100 Watt, that is, > 100 times too much
- 50 × R.E.A.L. 16023 DSP @ 200 MHz?
 - still > 10 × too much area and power
- Is SDR a realistic ambition ?

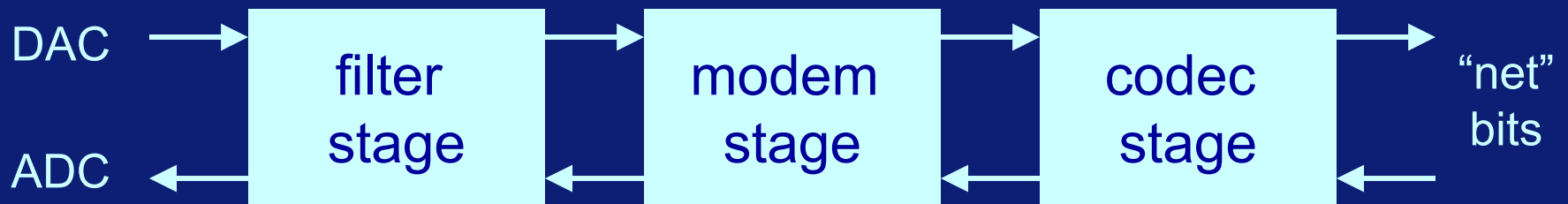


SDR baseband stages





SDR baseband stages: different functions



band limitation,
rate conversion

- filters

(de)modulation

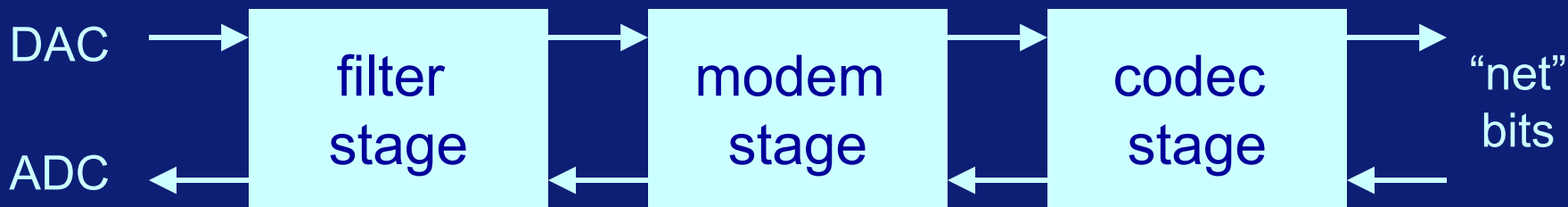
- equalization
- synchronization
- interference cancel.
- rake receiver
- QAM (de)mapping
- cordic, etc.

encoding/decoding

- (de)interleaving
- rate matching
- (de)puncturing
- viterbi, turbo dec.
- (de)multiplexing



SDR baseband stages: different characteristics



band limitation, rate conversion

- performance fixed by standard
- standards similar
- simple, regular

(de)modulation

- opportunity for differentiation
- standards different
- complex algorithms

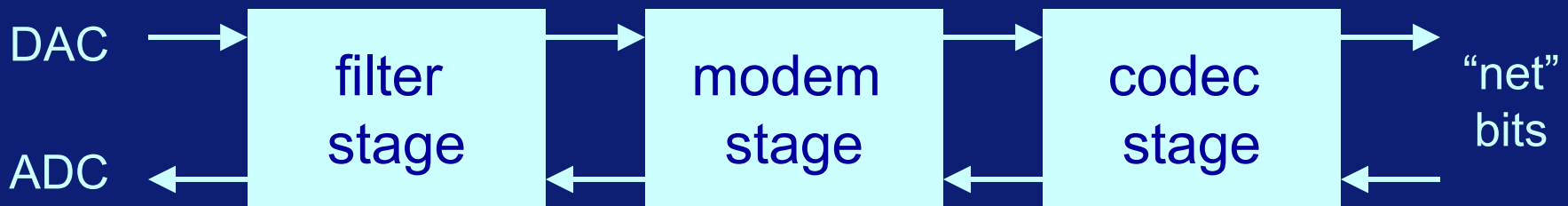
- lower bit-error rate
- wider coverage
- faster acquisition

encoding/decoding

- performance determined by standard algorithms
- standards similar
- complex algorithms



SDR baseband stages: different forms of flexibility



band limitation,
rate conversion

- tailored flexibility
- (re-)configurable filters

(de)modulation

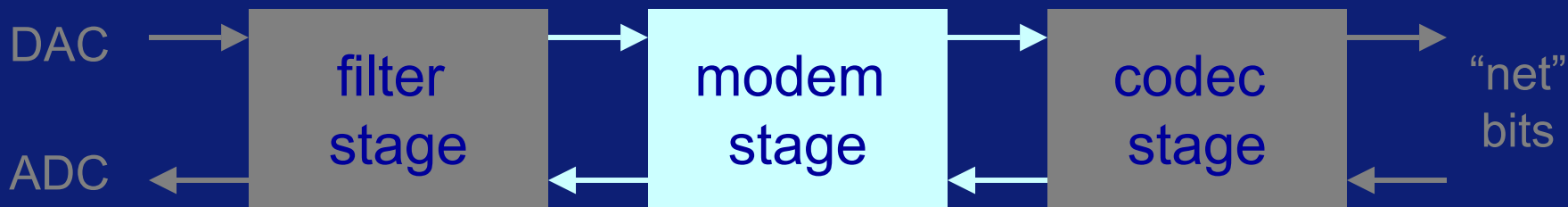
- SW flexibility
- 5-10 GIPS programmable processor

encoding/decoding

- tailored flexibility
- (re-)configurable viterbi/turbo + DSP and/or embedded FPGA



SDR baseband stages



- band limitation,
rate conversion
- tailored flexibility
 - (re-)configurable filters

- (de)modulation
- SW flexibility
 - 5-10 GIPS programmable processor

- encoding/decoding
- tailored flexibility
 - (re-)configurable viterbi/turbo + DSP and/or embedded FPGA



Characteristics of modern algorithms

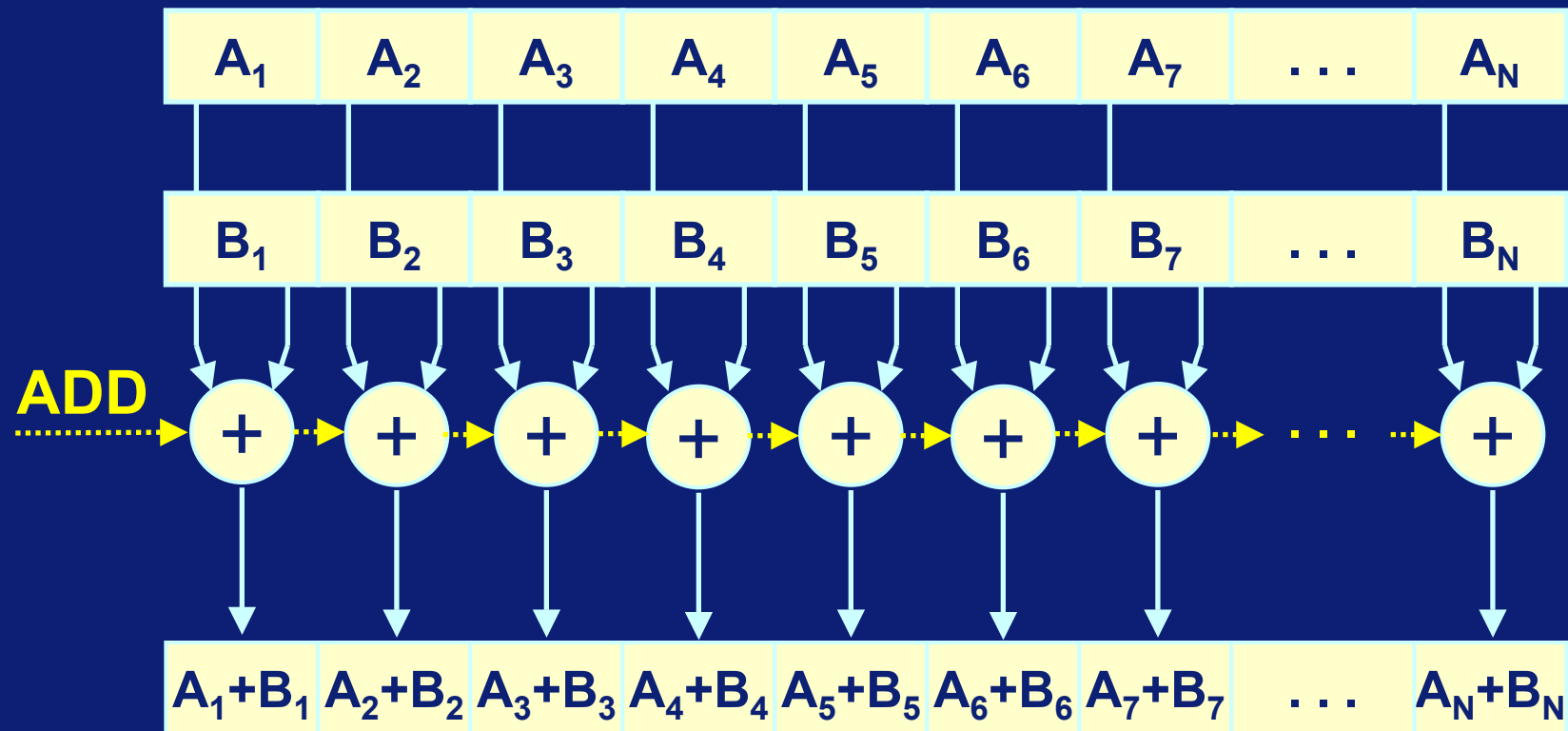
- variety of data types and word sizes.
- vast majority of operations are simple arithmetic ones.
- hundreds of operations per sample.
- high degree of locality of data.
- high degree of **data parallelism** available.
- control is recursive and simple.
- “data flow”: almost no data dependent control.

- However, **exceptions** make “**escapes**” essential!

⇒ **programmable vector processing**

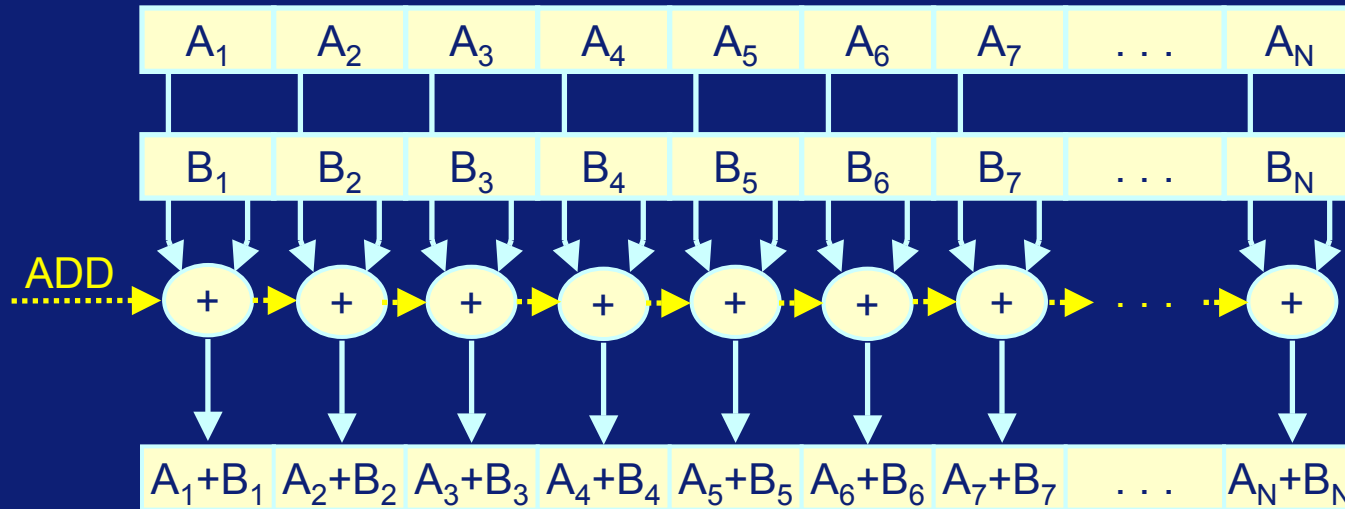


Vector Processing/SIMD





SIMD parallelism



SIMD = Single Instruction stream, Multiple Data stream

- + 1 {program memory, instruction decoder, L1 controller} (no/less SRAM fragmentation)
- + simple single-thread program model (e.g. task switch)
- ? less general (how much SIMD parallelism in appl)
- ? suffers from Amdahl's Law?

Efficient!

Flexible?

Research

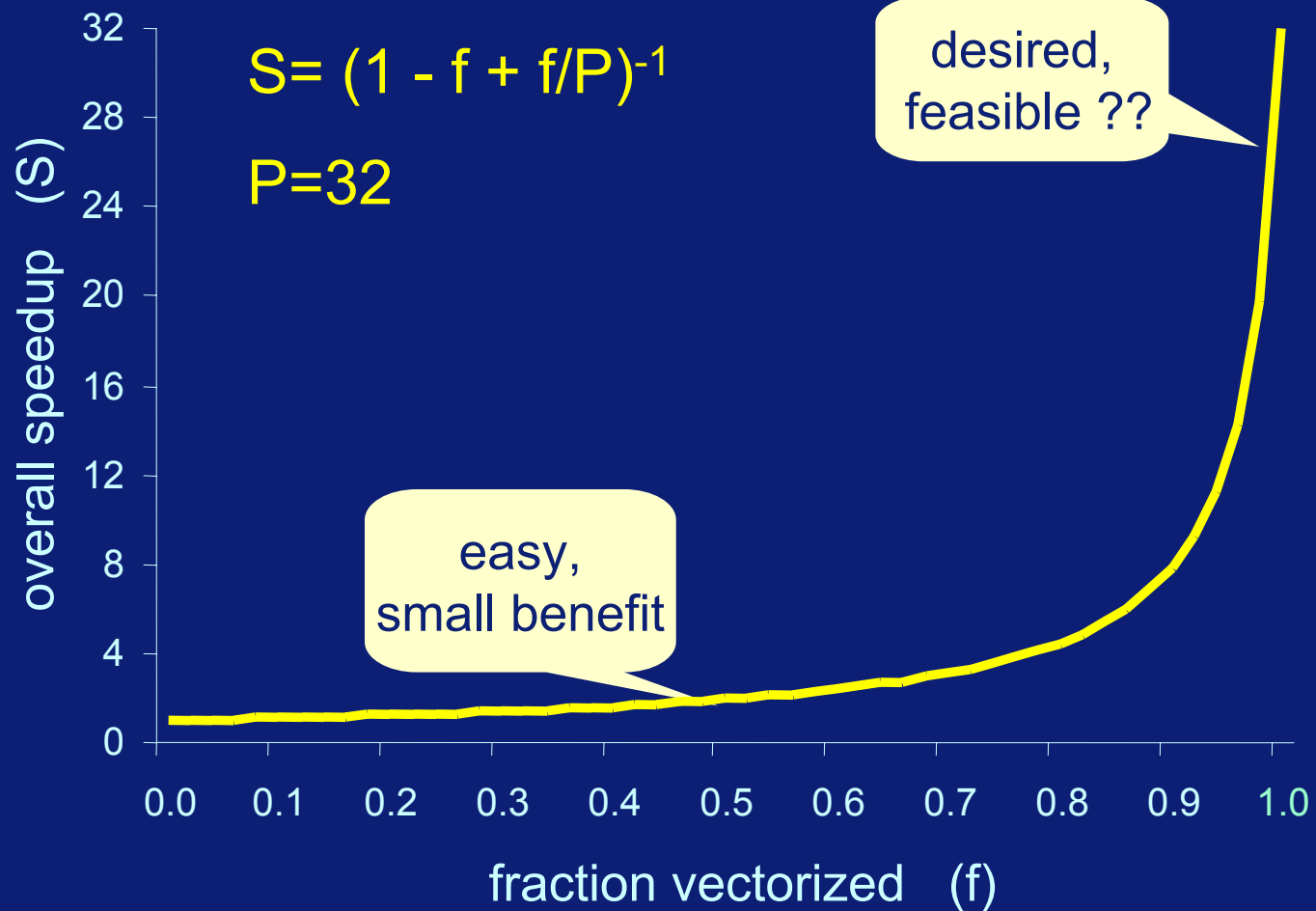


Many algorithms can be vectorized

- rake receiver
- UMTS acquisition
- cordic
- (I)FFT
- Fast Hadamard Transform
- OFDM symbol (de)mapping
- 16 QAM
- equalization
- symbol-timing estimation
- interference cancellation
- joint detection (TD-SCDMA)
- Viterbi decoder
- etc.
- RGB2YUV
- DCT
- SAD (incl. bilinear interpol.)
- motion estimation (feasible)
- video scaling
- vertical peaking
- disparity matching for mobile
- RGBd rendering
- color segmentation
- noise filtering (morphology)
- object filtering (i.e. aspect ratio)
- color interpolation
- etc.

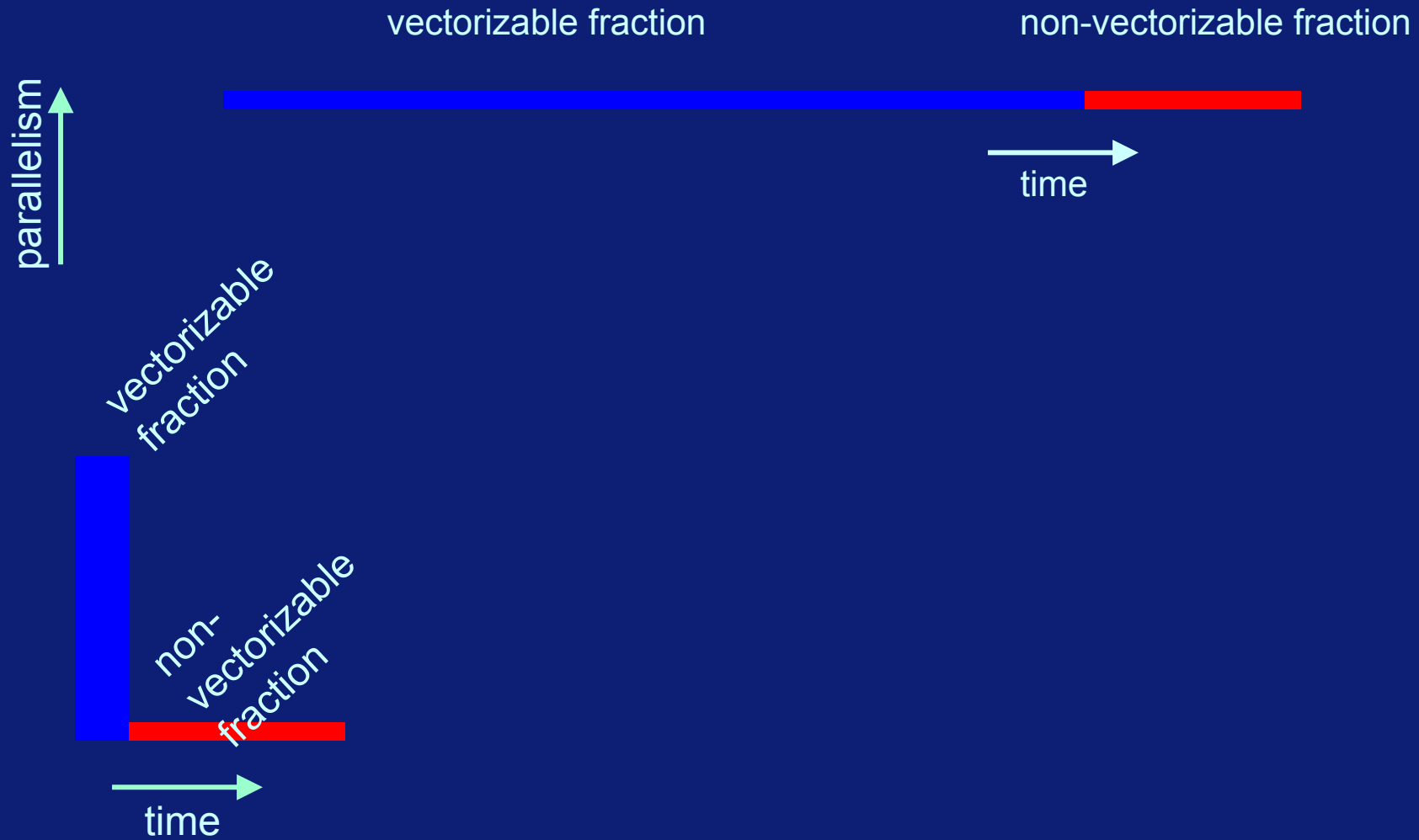
Performance typically scales
with vector size

Amdahl's Law





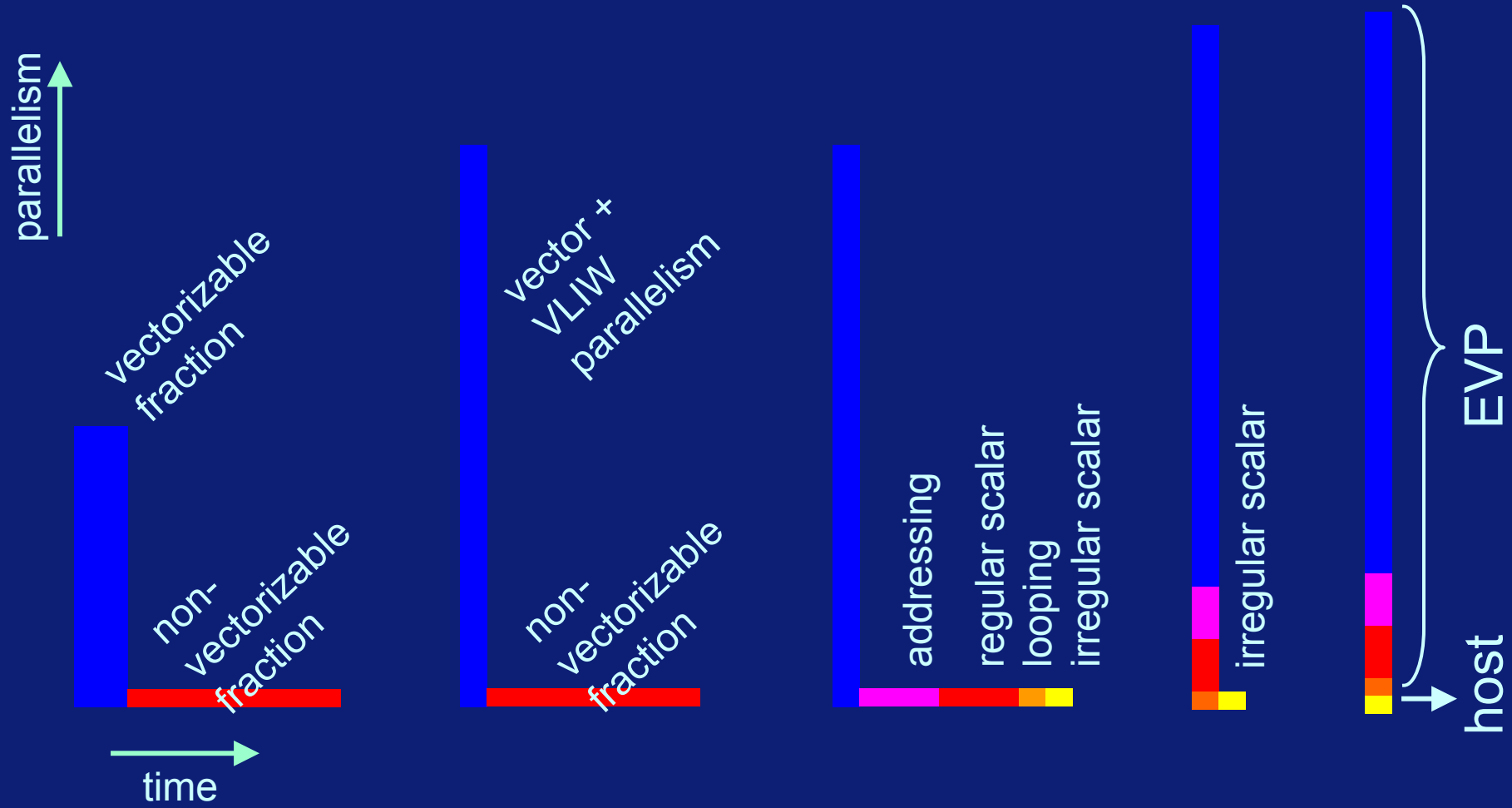
Amdahl's Law



Research



Countering Amdahl's Law





Requirements for an SDR vector processor

Raw performance:

- scalable SIMD width
- SIMD-wide memory access
1/fc throughput, low latency
- VLIW \times SIMD

+ to “counter” Amdahl’s Law:

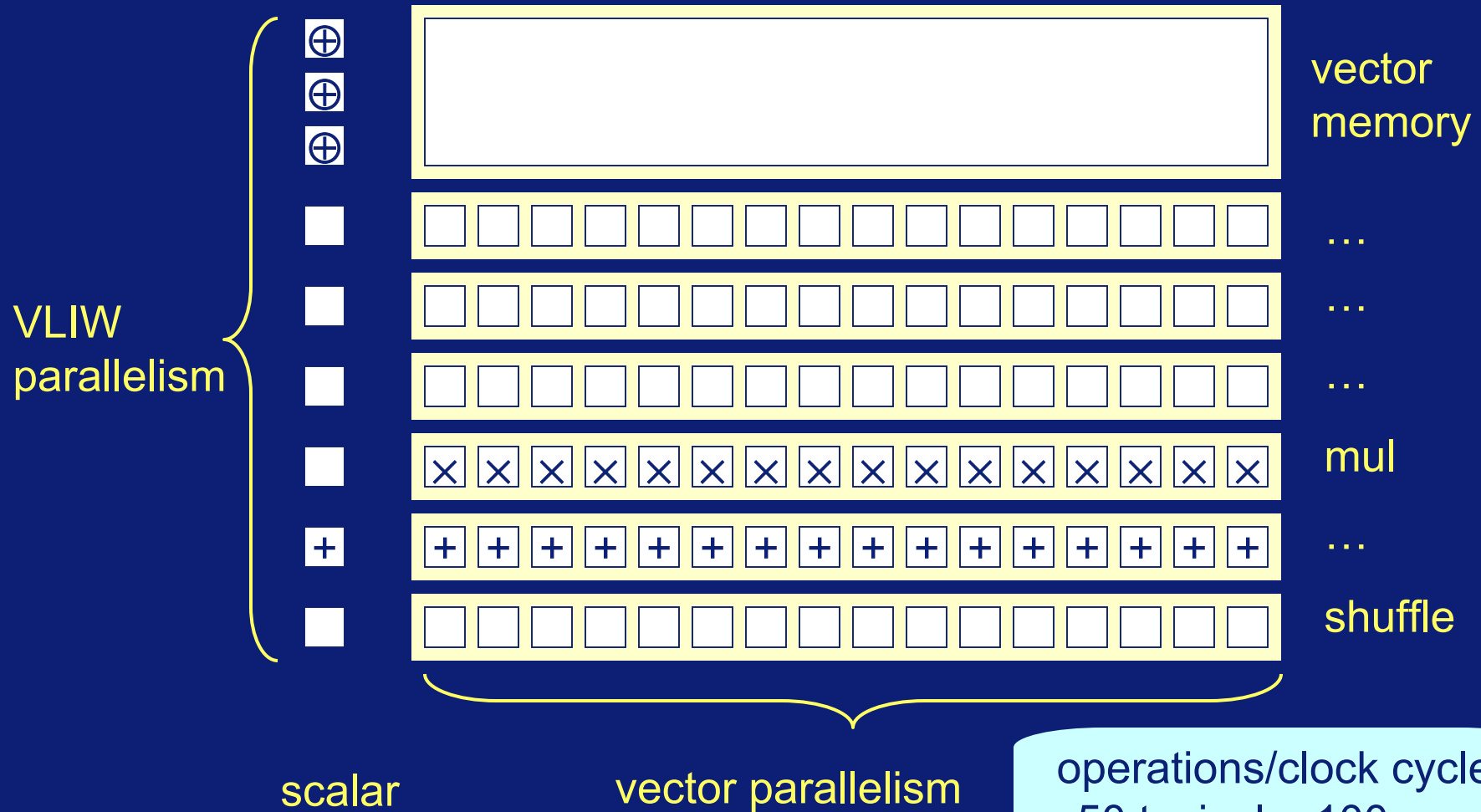
- scalar || vector ops
- address calculation units
- zero-overhead looping

Functional requirements

- intra-vector operations
 - e.g. sum of vector elms
 - segmented
 - rake, SAD, ..
- vector shuffle
 - permutation = special case
 - non-aligned VM access
 - FFT, Viterbi, SAD, ...
- (CDMA) code generation
 - 16 code-chips per cycle
 - multi-standard
 - CRC

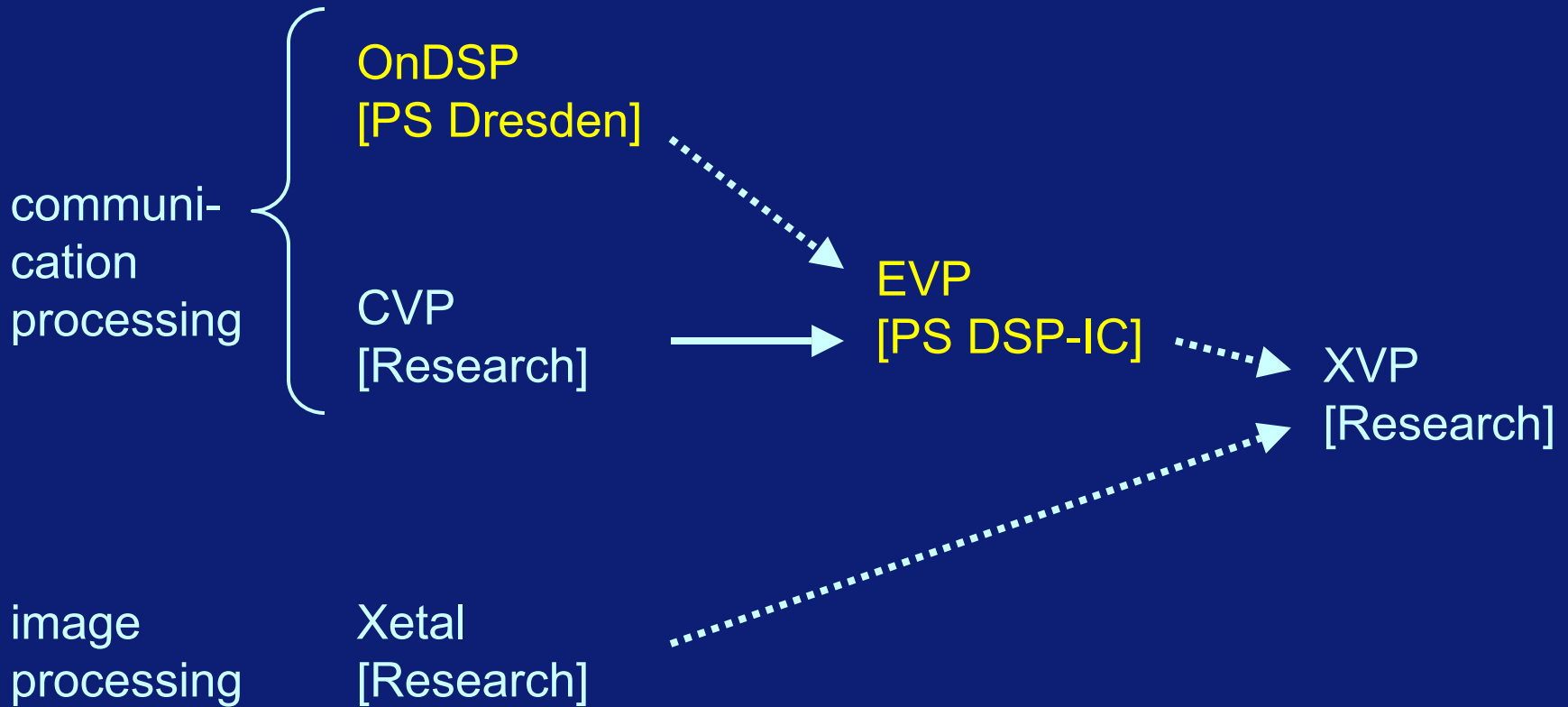


(scalar + vector ||) × VLIW ||

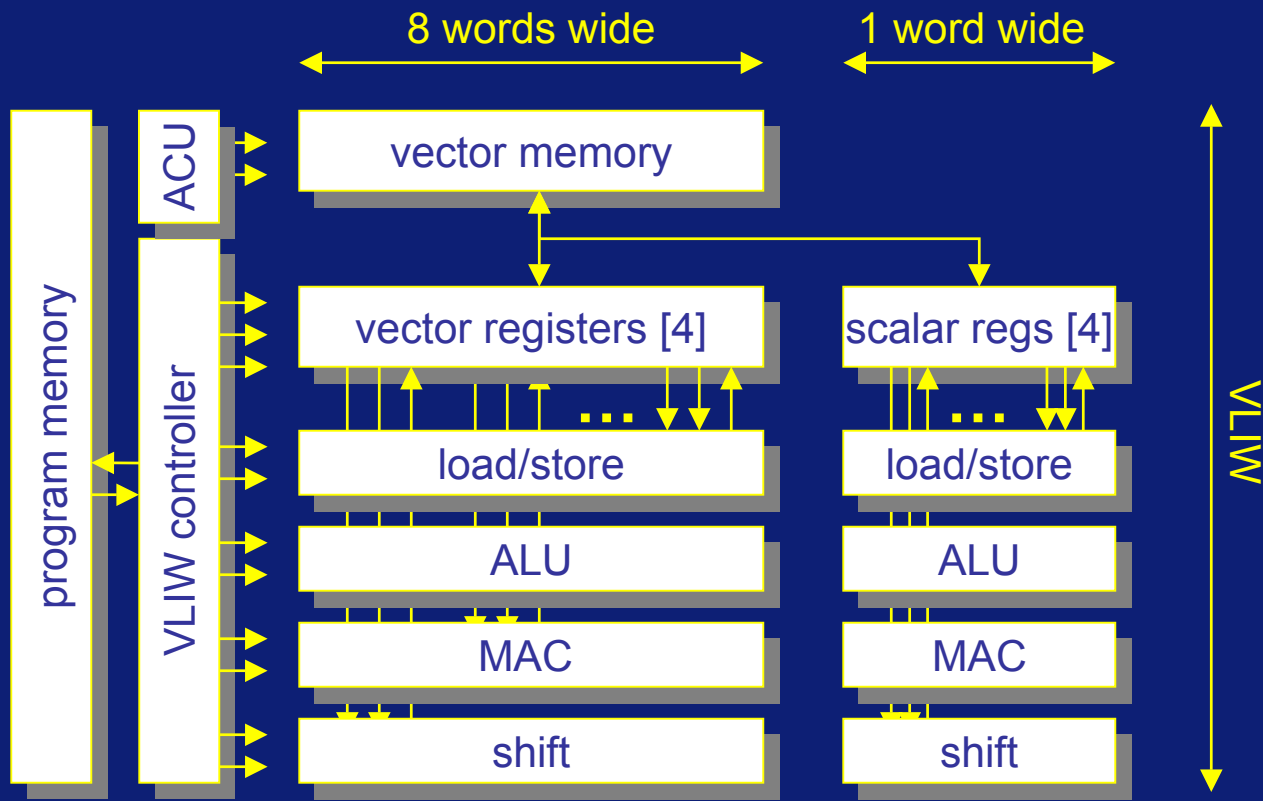




Vector processing @ Philips



OnDSP Architecture [existing product]



OnDSP
in 120 nm CMOS:

- 250 k gates
- 1.5 mm²
- 160 MHz (worst case)
- 0.8 mW/MHz typ memory configuration

- Basis for 802.11a/b/g (products)
- Considered for DVB-T/H, Wimax

Research



FFT performance [block size = 64]

processor		source		cycles	relative
EVP ₁₆	opt	Philips		64	1.0
OnDSP	opt	Philips		160	2.5
Tigersharc	opt	AMD		206	3.2
VIRAM	opt	Berkeley	Kozyriakis	357	5.6
TMS320C6203	opt	TI	eembc	646	10.1
R.E.A.L. RD16023	opt	Philips	bdti	678	10.6
Altivec MPC7447	opt	Motorola	eembc	956	14.9
Carmel 10xx	otb	Infineon	eembc	5568	87.0
AMD K6-2E+/ACR	otb	AMD	eembc	10751	168.0

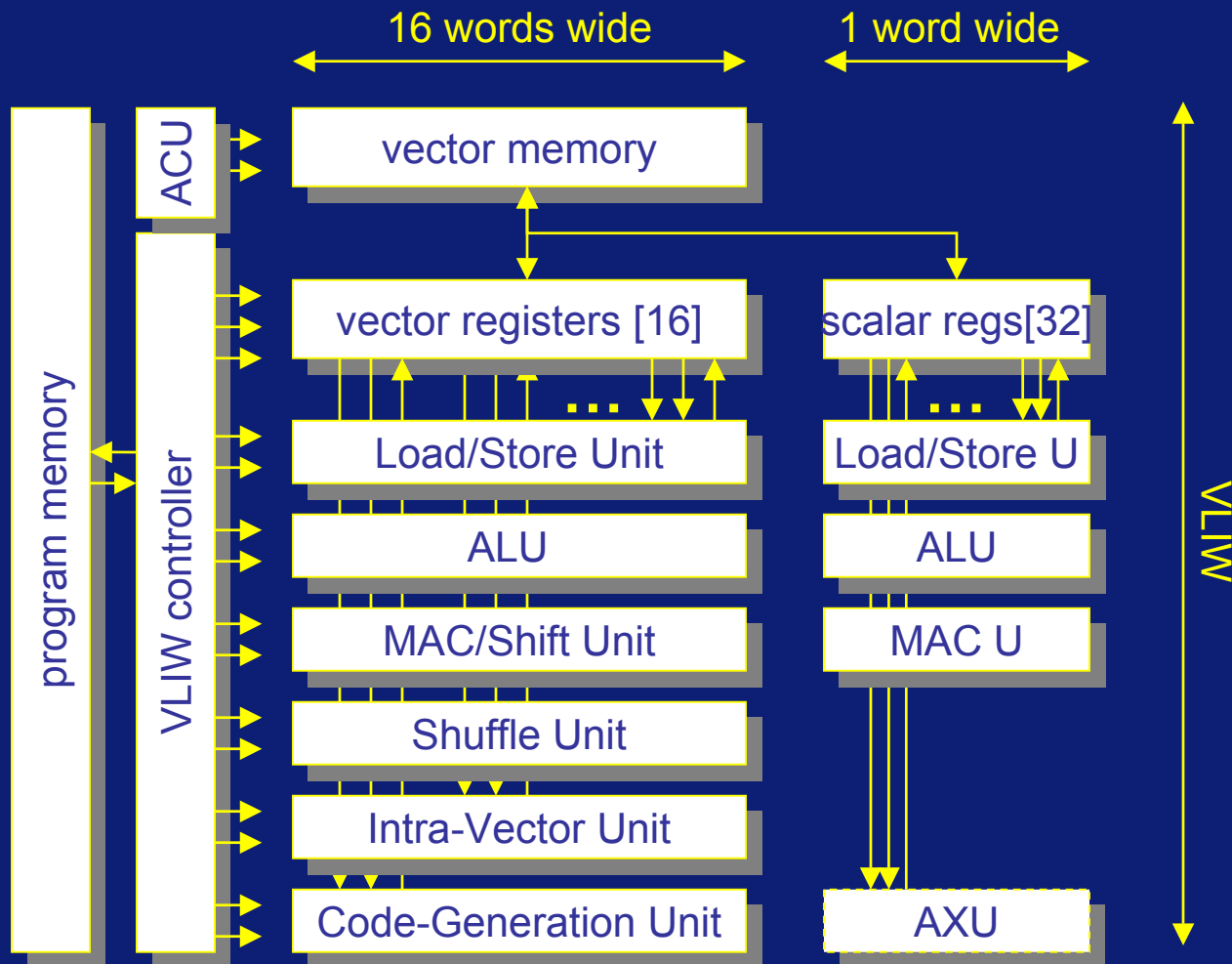
Research



802.11a (de)modulator load on OnDSP

OnDSP load (measured) (de)modulator task	TX [cycles]	RX [cycles]	
symbol (de)mapping	35	35	per OFDM symbol [4 μsec.]
pilot generation	55		
pre (post) scaling	35	35	
tracking		36	
channel correction		39	Equivalent to • 110 MHz (OnDSP, product)
OFDM (de-) mapping	35	35	
afc(I) FFT	160	160	• < 55 MHz (EVP estimate)
TS frequency shift	40		
phase-error correction		39	
CP insertion/removal	35		
control code	40	40	
overall peak load	435	414	

EVP architecture



EVP₁₆
In 90 nm CMOS:

- 450 k gates
- ≈ 2 mm²
- 300 MHz (worst case)
- 0.5 mw/MHz core only
- 1 mW/MHz typ memory configuration

(Vectorizing) compilers

identify loops and array operations

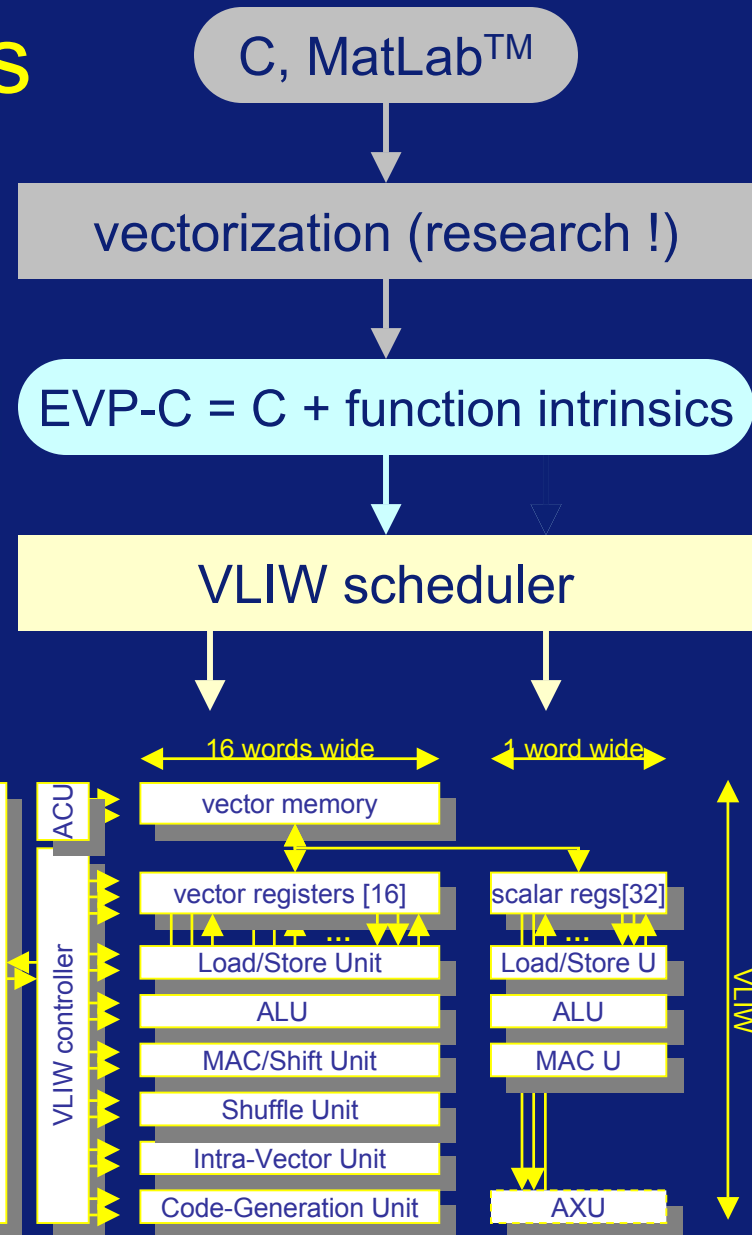
e.g. $a = \text{vmac}(a, b, c)$ (no float)

register allocation, scheduling

“Compiling for vector machines is the most mature of the parallel compiler technologies.”

Michael Wolfe – High Performance Compilers for Parallel Computing. 1996 by Addison-Wesley Publishing Company, Inc

Research



Example code: Inner part of Hadamard transform

```
extern vbool sign_data[];
void hadamard_intra(vfrac16 * in, vfrac16 * out)
{
    vbool * sign_ptr = sign_data;
    vbool sign0,sign1,sign2;
    vfrac16 = w_0, w_1, w_2, w_3;
    sign0 = *sign_ptr++;
    sign1 = *sign_ptr++;
    sign2 = *sign_ptr;
    for (i=0; i<N; i++)
    {
        w_0 = *in++;
        w_1 = vshuf(bfy32,w_0); // butterfly shuffle pattern 1
        w_1 = vcas(sign0,w_1,w_0); // conditional add/subtract ( * ±1)
        w_2 = vshuf(bfy64,w_1);
        w_2 = vcas(sign1,w_2,w_1);
        w_3 = vshuf(bfy128,w_2);
        w_3 = vcas(sign2,w_3,w_2);
        *out++ = w_3;
    }
}
```

Variable mapped
on register by tool

Instructions
scheduled by tool

Operations
working on
vectors

Note:
preliminary
syntax, exact
syntax still
under
discussion





EVP-C and tools

EVP-C:

- ANSI-C ... maps on scalar FUs
- + vector types + vector function intrinsics

Tools:

- C++ host emulation library
- compiler: VLIW scheduling of scalar + vector operations
 - ELF executable and linking format + DWARF debug format
- linker
- cycle-true bit-true simulator
- integrated debugger
- profiler



EVP₁₆: load per UMTS rake-finger

processor	load/rake finger [MHz]	arithmetic resources [complex arithmetic]
EVP ₁₆	0.5	16 [MAC+ ALU + PN gen.]
TigerSharc	1	2*8 [MAC+ ALU]
UMTS DP	25	1 MAC+ ALU + PN gen.]
TI C62	40	
R.E.A.L. RD16023	100	2 MAC + 2 ALU
Carmel	125	2 MAC + ALU
TI C54x	300	1 MAC/ALU

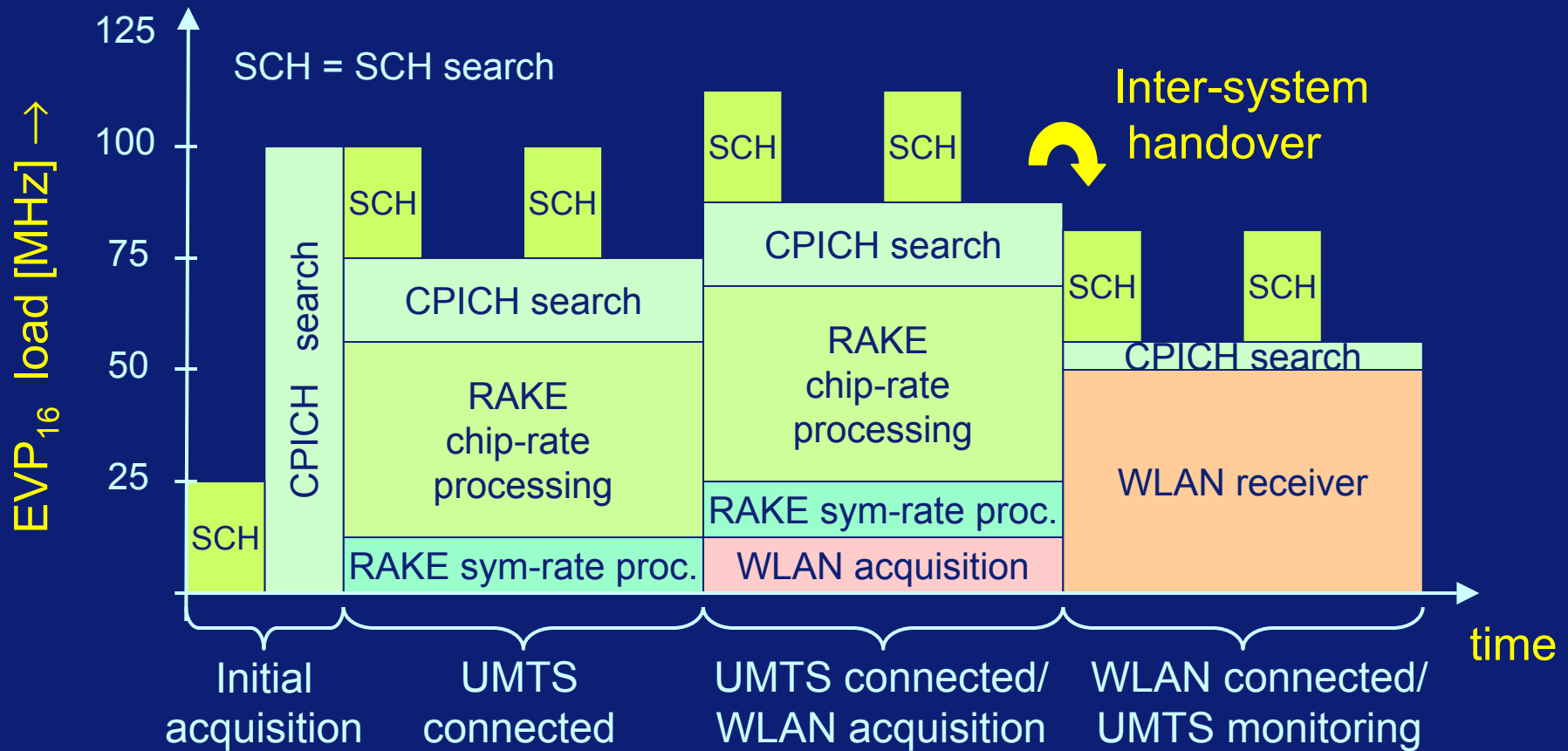
- UMTS may require more than 50 rake fingers!



UMTS/HSDPA (de)modulator load on EVP₁₆

EVP ₁₆ load (estimates)	[MHz]	scenarios				Scenarios:
UMTS (de)modulator task	1	2	3	4		
PSCH search (Golay)	18	18	18	18	1.	idle (multi-cell synchronization)
CPICH search	98	17	17	17		
CPICH despreading		4	22	33	2.	R'99 connected ("flat fading")
CPICH symbol rate		1	1	2		
DCH despreading		3	16	16	3.	R'99 connected ("scattering")
DCH symbol rate		1	6	6		
HS-SCCH despreading				15	4.	R'99 connected + HSDPA ("scattering")
HS-SCCH symbol rate				1		
HS-DSCH despreading				12		
HS-DSCH symbol rate				22		
Overall peak load	116	44	80	142		
overall average load	4	28	64	126		

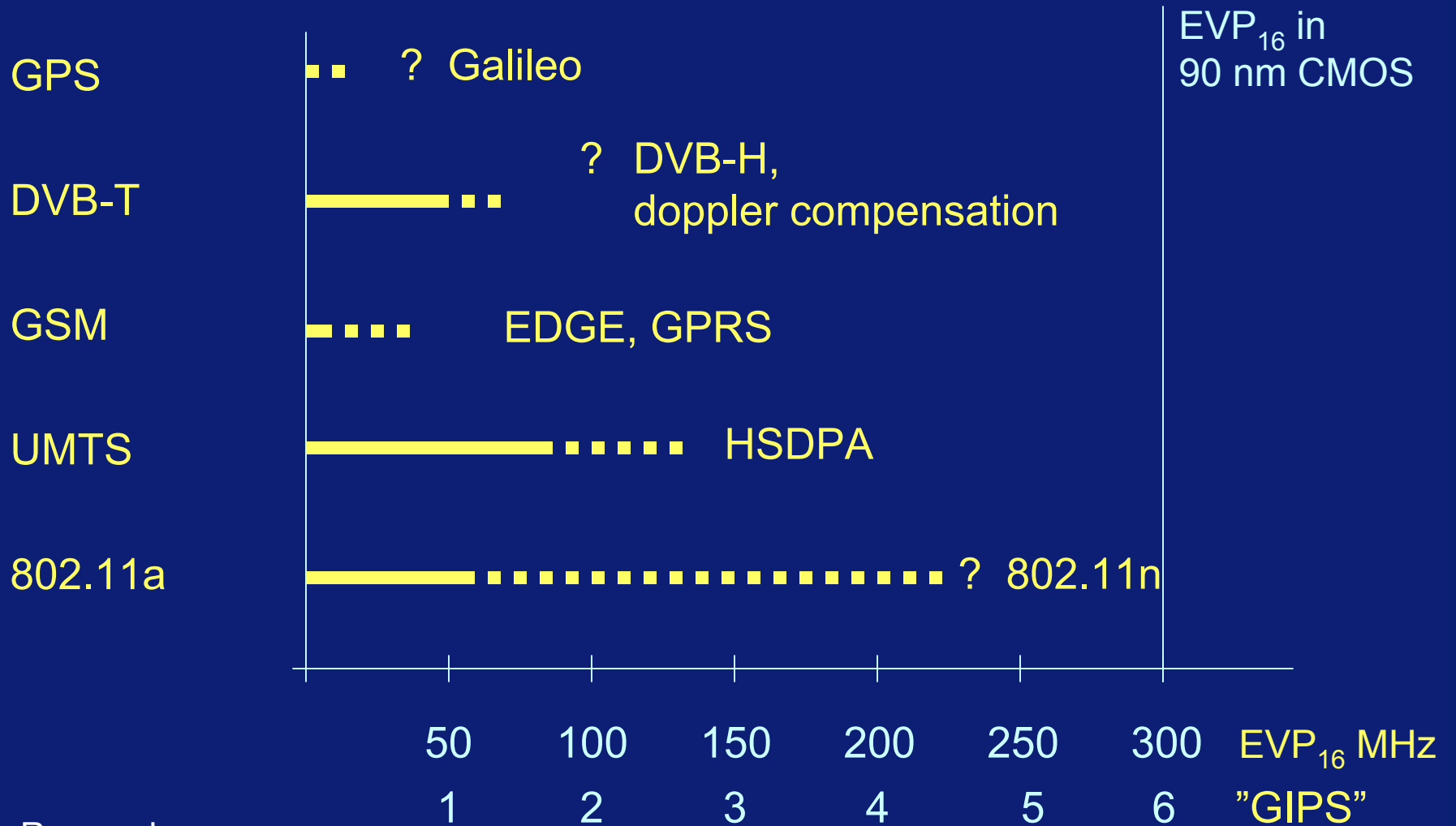
Software saves silicon area!



resource sharing: both intra-standard and inter-standard!

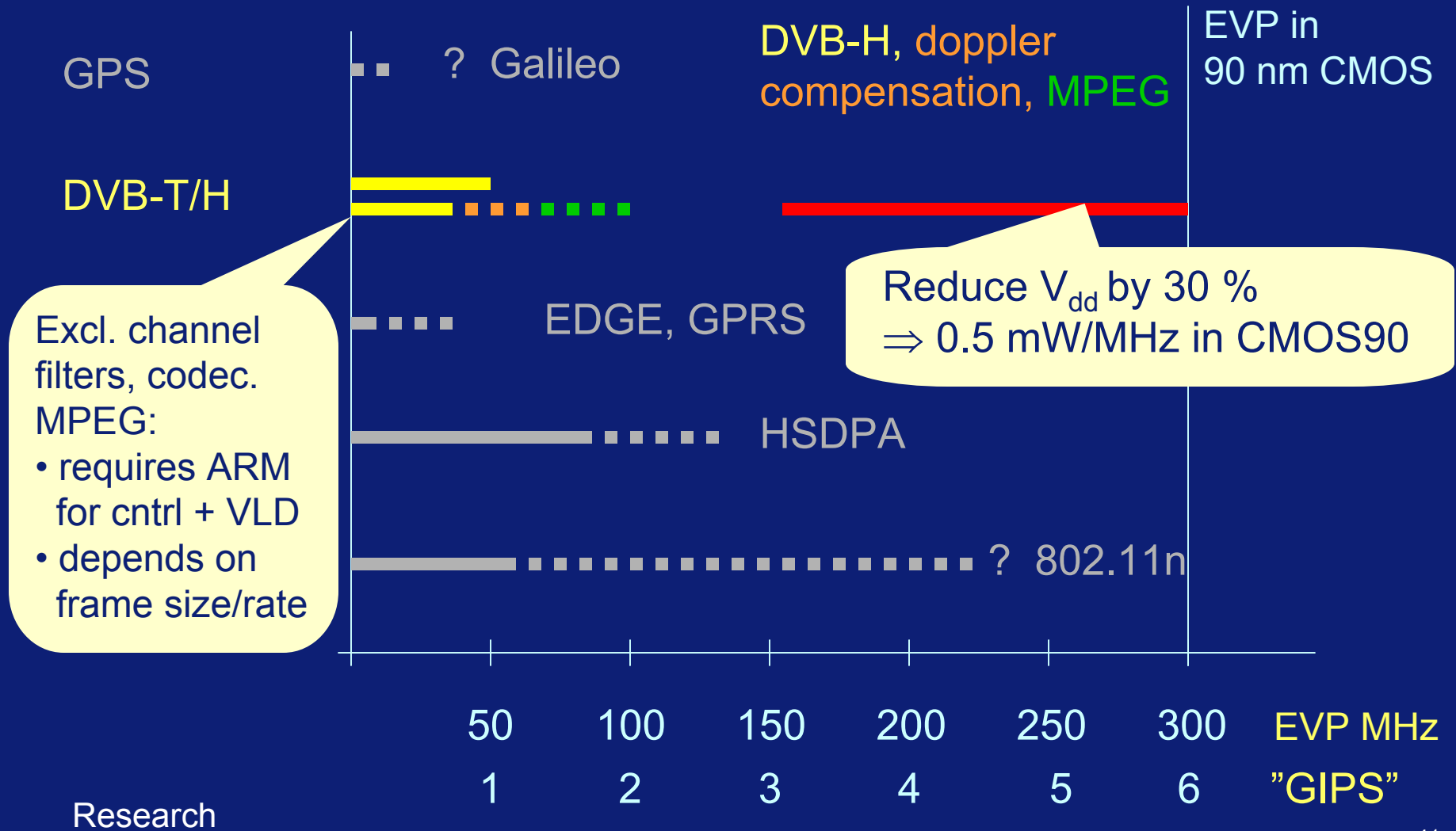


“Modem” peak load on EVP₁₆ (estimates)



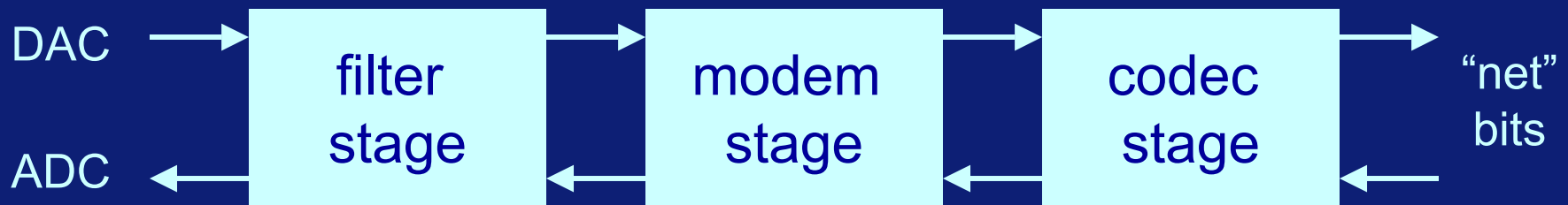


DVB Modem: use of EVP_{16} headroom (estimates)





SDR baseband stages: different forms of flexibility



band limitation,
rate conversion

- tailored flexibility
- (re-)configurable filters

(de)modulation

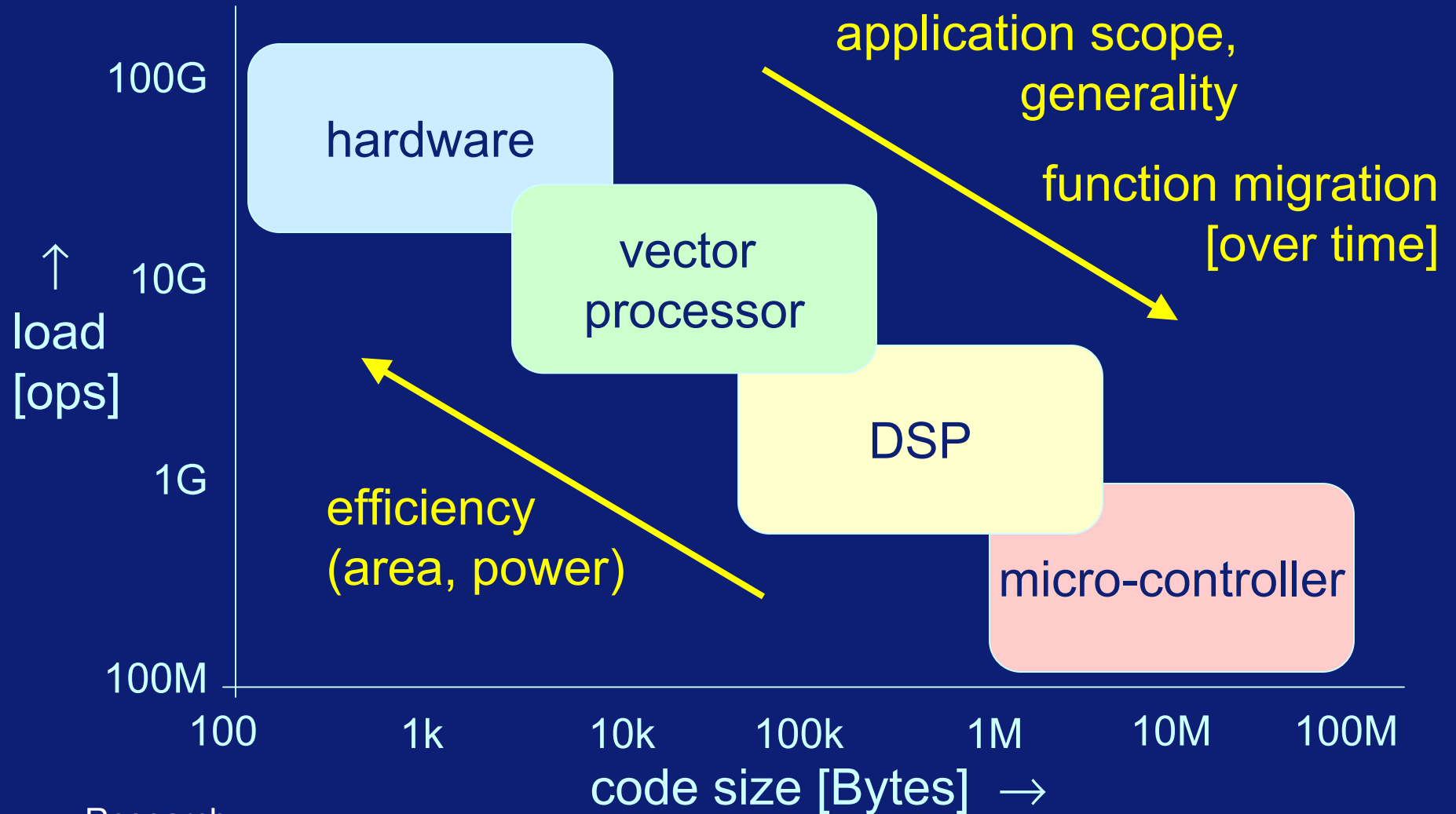
- SW flexibility
- 5-10 GIPS programmable processor

encoding/decoding

- tailored flexibility
- (re-)configurable viterbi/turbo + DSP and/or embedded FPGA



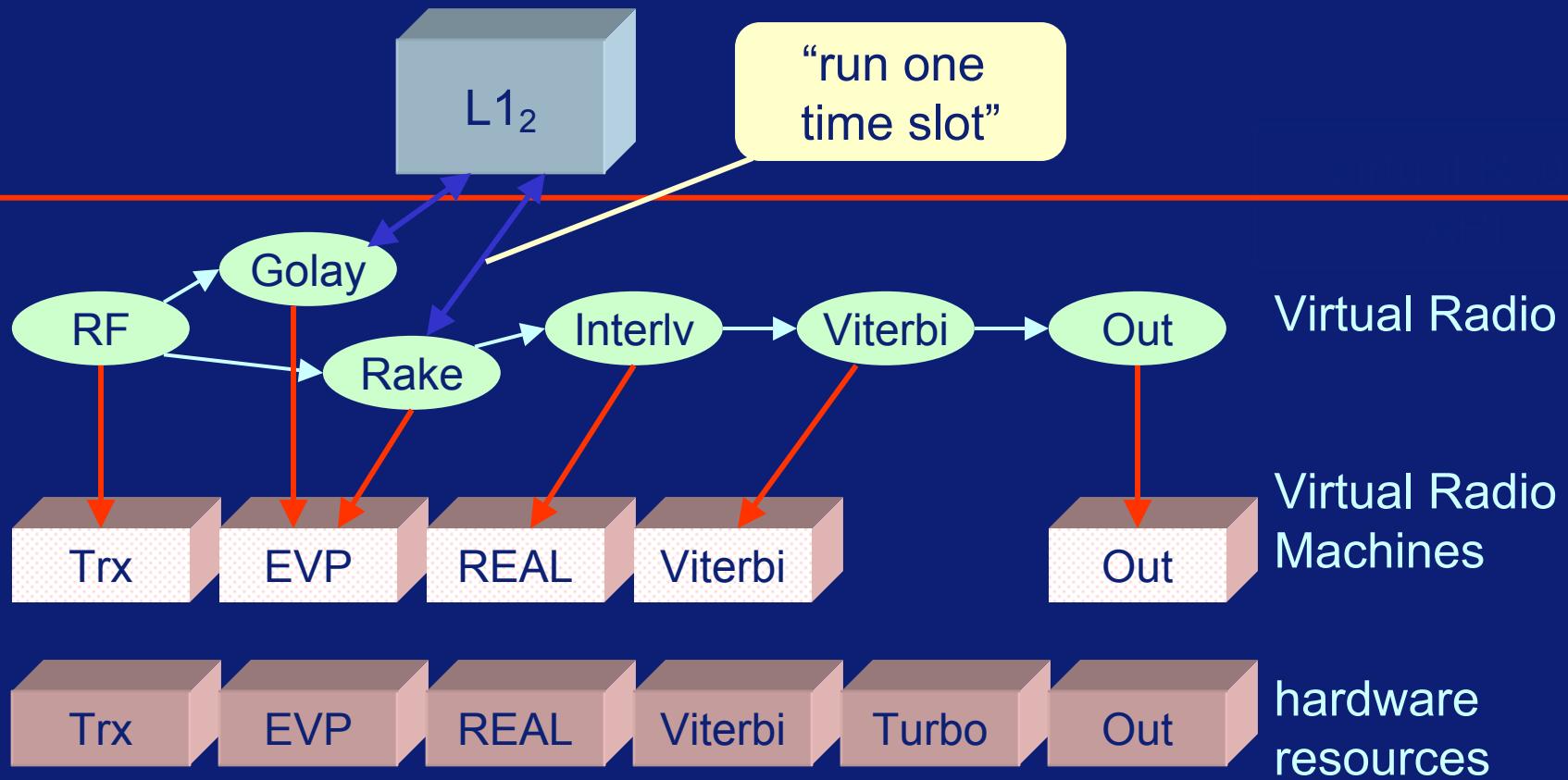
System = HW + vector processor + DSP + GP



Research

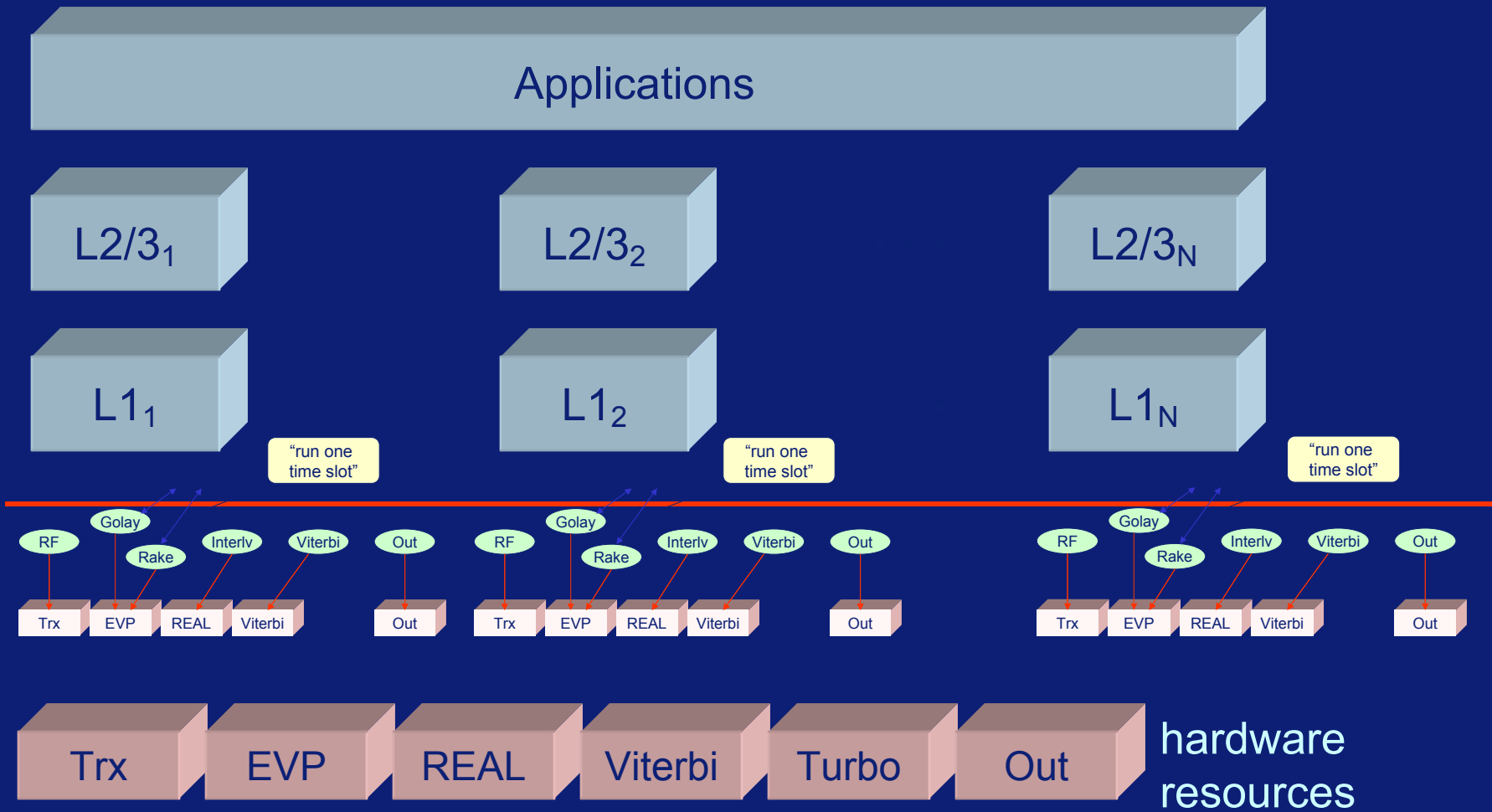


Virtual radio, UMTS example





Virtual radio: simultaneous operation





Conclusion

- Uncertainties on combinations of standards {cellular, broadcast, connectivity, positioning} and on their evolution
⇒ Software-Defined Radio.
- Software flexibility must be applied only when/where valuable: for time-to-market, product differentiation, and cost reduction.
⇒ (mainly) modem stage.
- Programmable vector processing is a key-enabler for SDR: it delivers the required “flexible” GOPSes (with headroom), saves silicon area, and is low power.
- EVP has been prepared for cellular, broadcast, and connectivity standards with UMTS/HSDPA as lead application.
- Current research: SW-modem → add SW-codec → SDR.

