Real-Time and Embedded Systems ?
- Chapter 1 & 2 -

Definition is not clear!

Let’s take a look at “examples”

- Cell phones, PDAs
- Digital cameras
- Microwave ovens
- Network adaptor box (e.g., ISDN adaptor)
- Multimedia systems such as DVR, VOD server, etc
- Factory process control
- Radar systems
- Avionics

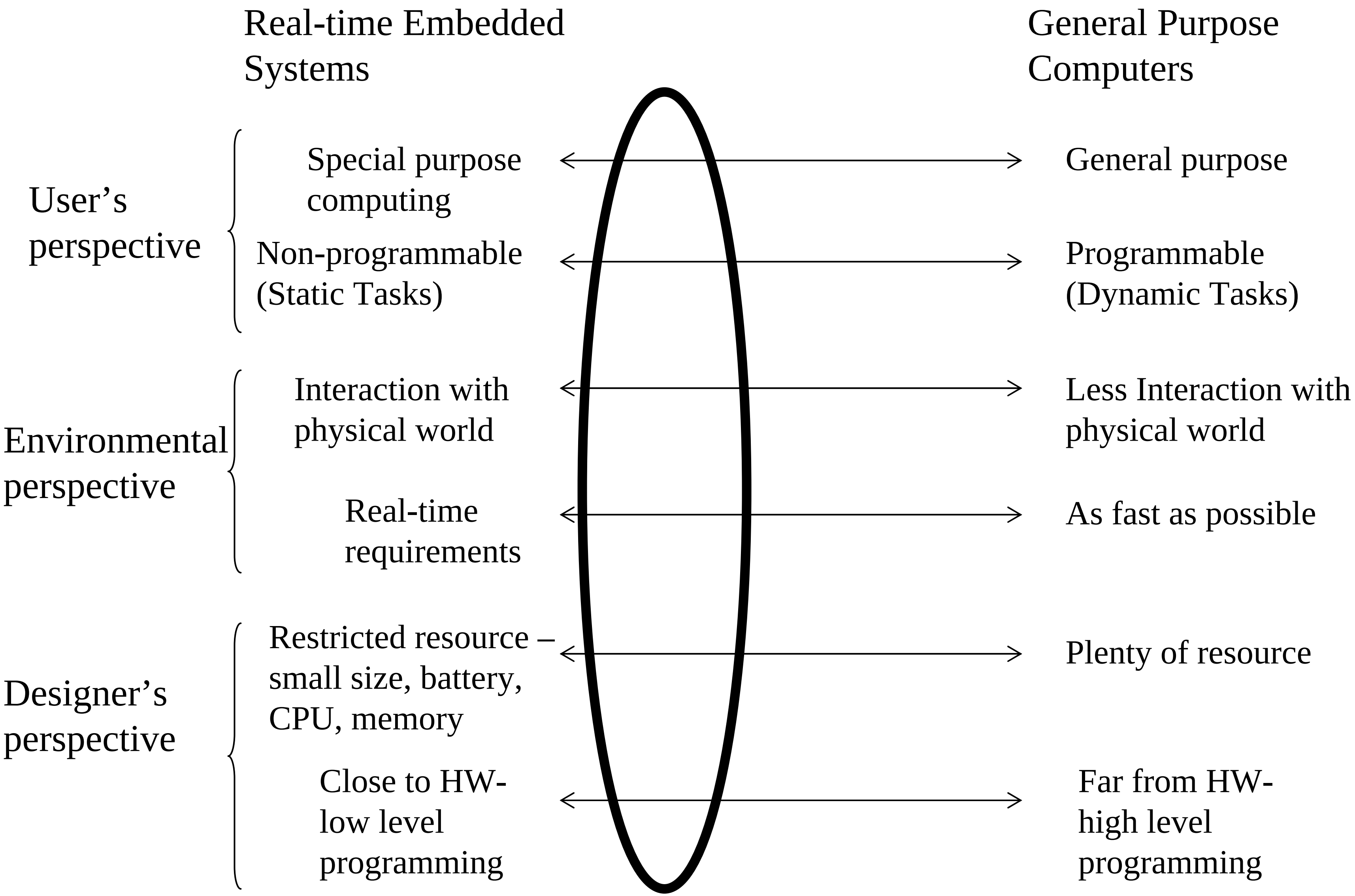
What are embedded systems?

- Computing system embedded in a larger system
    - Provide computing for a system with special purpose (cf. general purpose computing)
    - Computing itself is hidden from the user
        - computing is not the ultimate goal
        - computing serves for real mission (ECU of cars)
    - Usually perform static tasks - predetermined
    - Usually small to be embedded
    - Battery operated sometimes
    - Usually have “real-time” constraints

What are real-time systems?

- Computing system whose specification includes both logical and temporal correctness requirements
  - *Logical Correctness*: produces correct outputs
  - *Temporal Correctness*: produces outputs at the right time
  - Usually “embedded”

Real-Time Embedded Systems

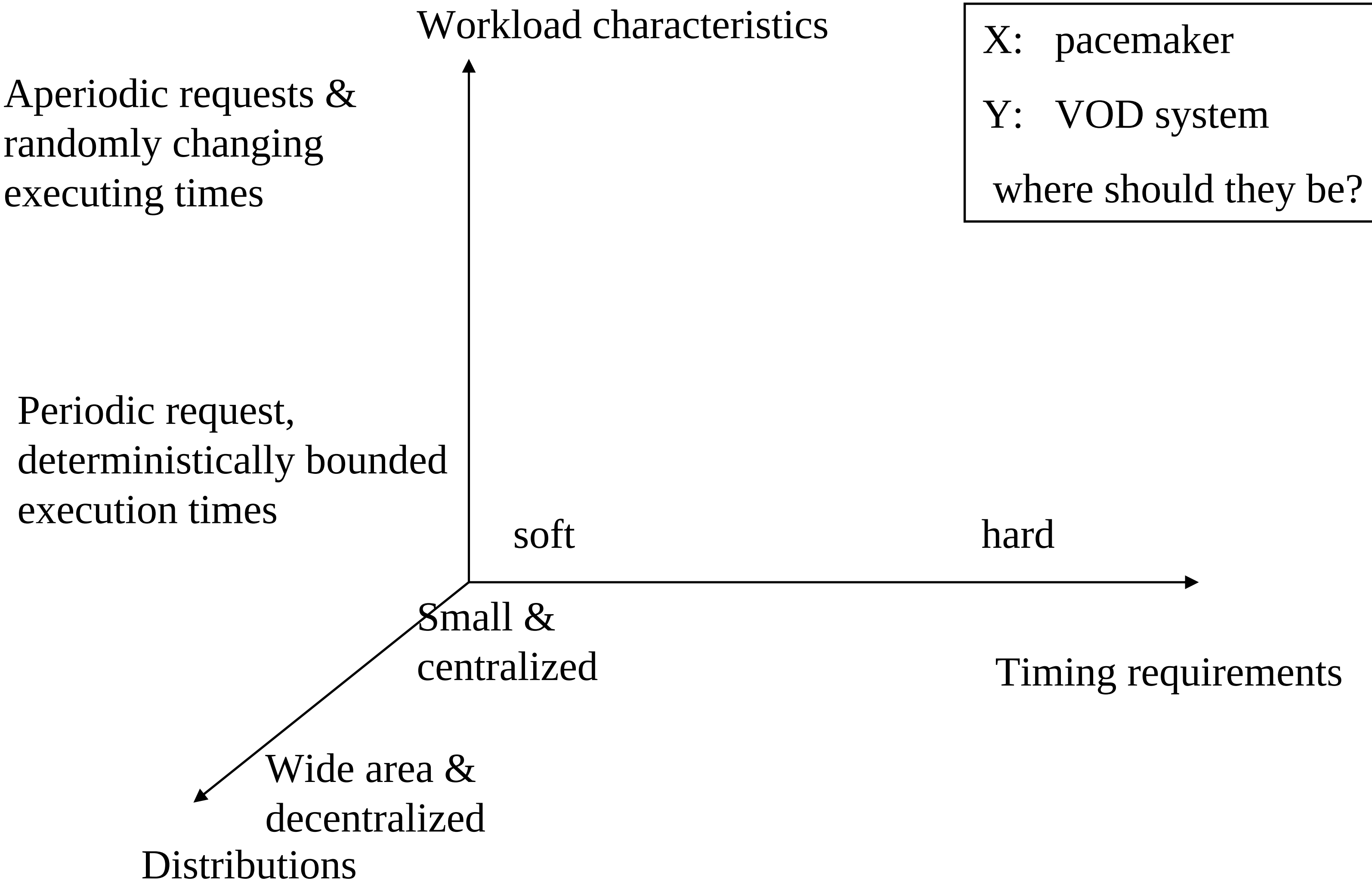


	Real-time Embedded Systems	General Purpose Computers
User's perspective	Special purpose computing	General purpose
	Non-programmable (Static Tasks)	Programmable (Dynamic Tasks)
Environmental perspective	Interaction with physical world	Less Interaction with physical world
	Real-time requirements	As fast as possible
Designer's perspective	Restricted resource – small size, battery, CPU, memory	Plenty of resource
	Close to HW- low level programming	Far from HW- high level programming

Taxonomy of Real-time Embedded Systems

- Hard or Soft timing constraints
  - Missing HARD deadlines will be catastrophic
  - Missing SOFT deadlines will degrade the satisfaction
- Static or Dynamic workload
  - STATIC tasks are given at the design phase
  - DYNAMIC tasks created at the run time
- Centralized or Decentralized
  - Centralized small embedded systems
  - Decentralized networked embedded systems

Three Dimensions



Workload characteristics

Aperiodic requests & randomly changing executing times

Periodic request, deterministically bounded execution times

soft

hard

Timing requirements

Small & centralized

Wide area & decentralized

Distributions

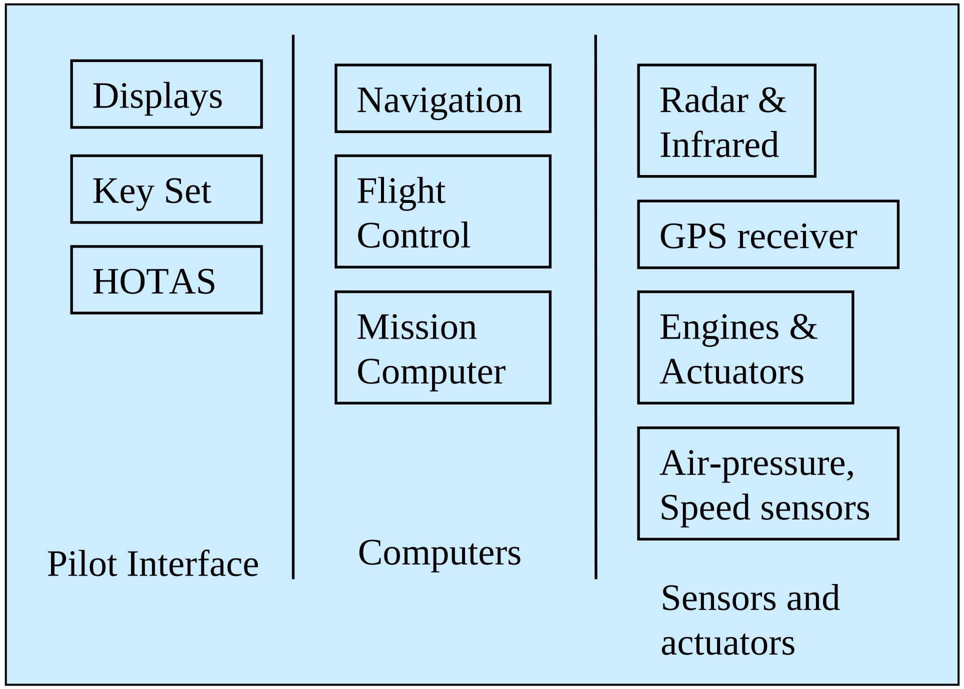
X: pacemaker

Y: VOD system

where should they be?

An Example Real-Time Embedded System

HW Components



Displays

Key Set

HOTAS

Pilot Interface

Navigation

Flight Control

Mission Computer

Computers

Radar & Infrared

GPS receiver

Engines & Actuators

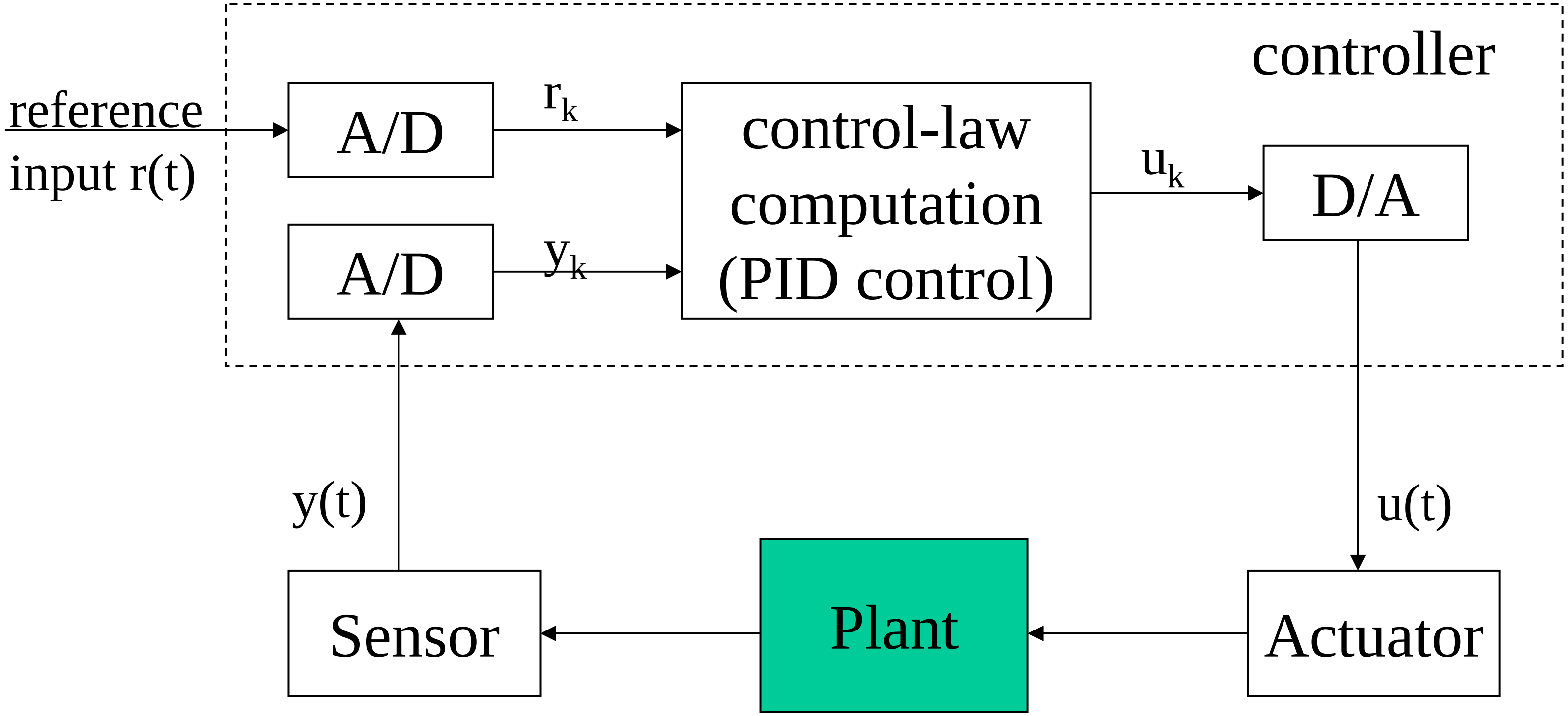
Air-pressure, Speed sensors

Sensors and actuators

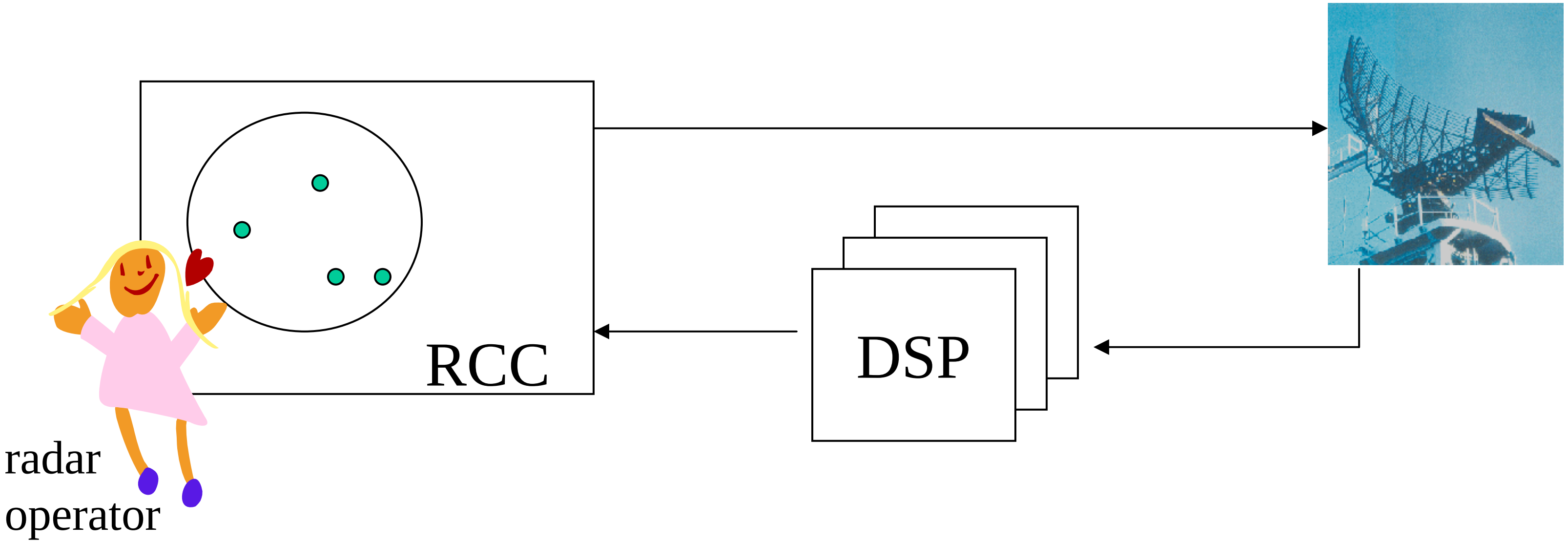
SW Tasks

Tasks	**Periods**
Display Update	200 ms
Tracking Target Update	100 ms
Navigation Update	60 ms
Data bus polling	40 ms
Collision Warning Receiver	25 ms

Control Applications



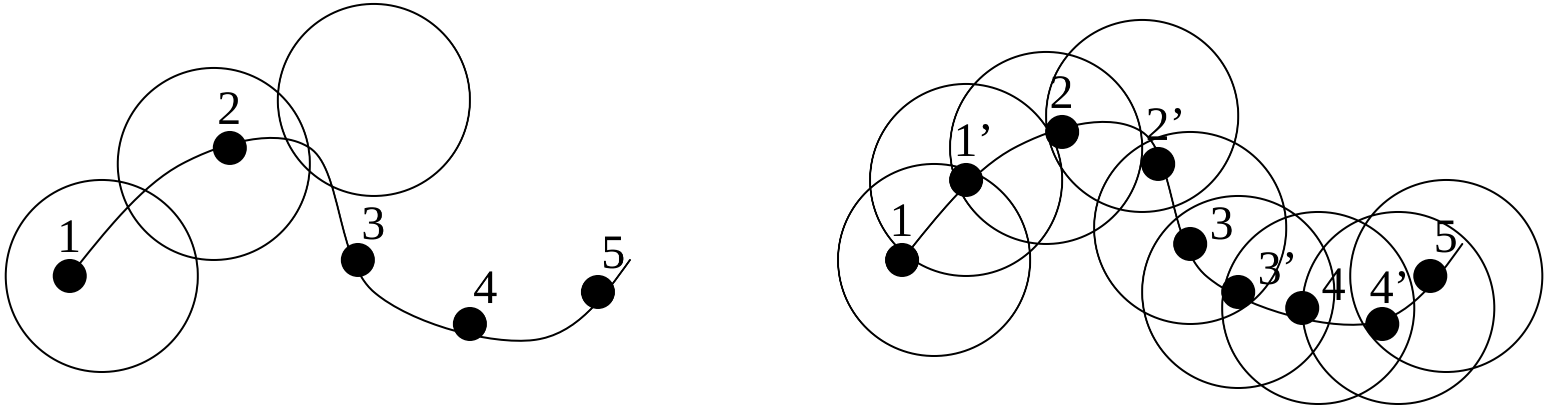
Radar Systems



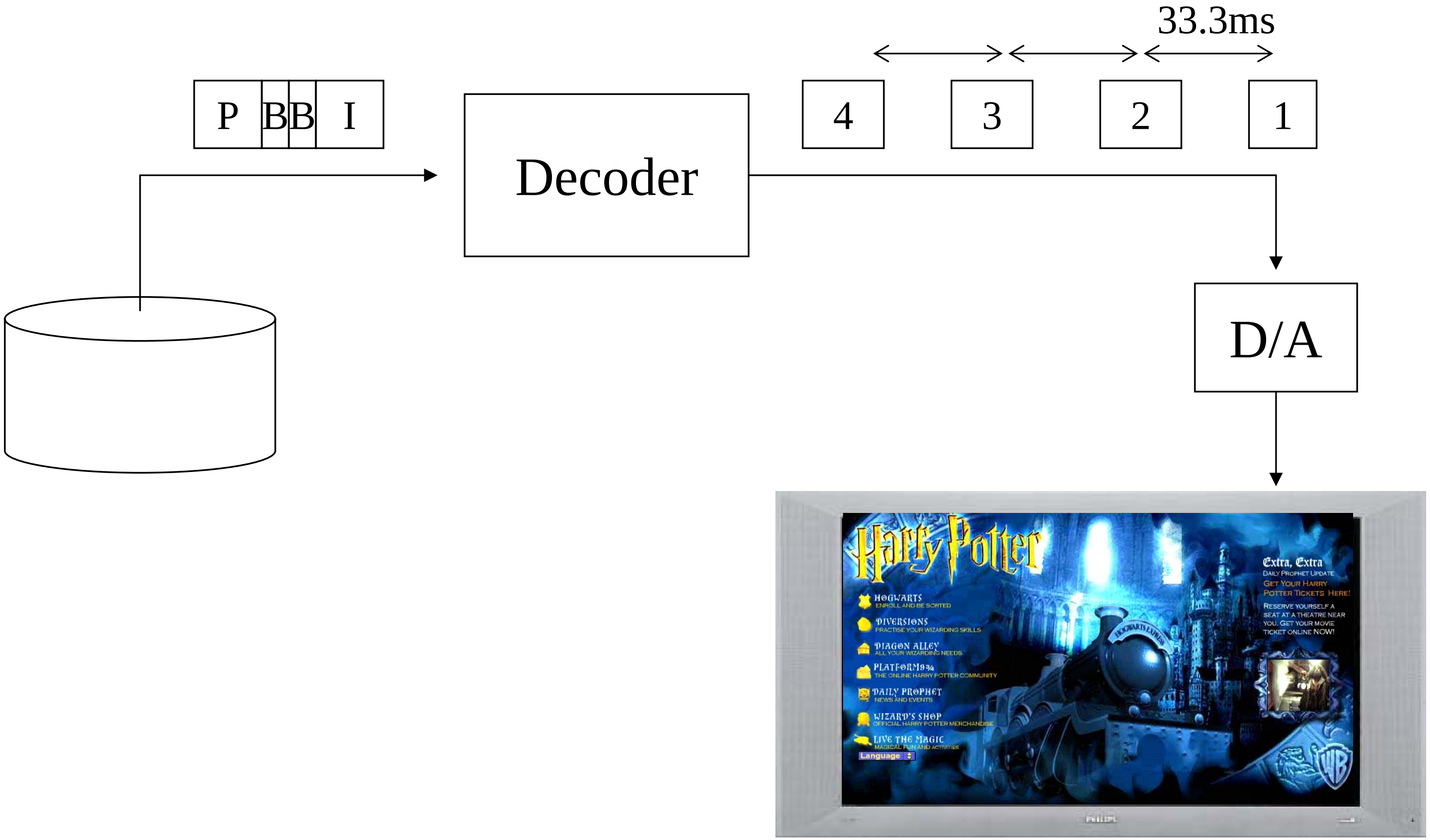
RCC

DSP

radar operator



VOD system



33.3ms

P BB I

Decoder

4 3 2 1

D/A

Harry Potter
HOGWARTS
ENROLL AND BE SORTED
DIVERSIONS
PRACTISE YOUR WIZARDING SKILLS
DIAGON ALLEY
ALL YOUR WIZARDING NEEDS
PLATFORM9¾
THE ONLINE HARRY POTTER COMMUNITY
DAILY PROPHET
NEWS AND EVENTS
WIZARD'S SHOP
OFFICIAL HARRY POTTER MERCHANDISE
LIVE THE MAGIC
MAGICAL FUN AND ACTIVITIES
Language
Extra, Extra
DAILY PROPHET UPDATE
GET YOUR HARRY POTTER TICKETS HERE!
RESERVE YOURSELF A SEAT AT A THEATRE NEAR YOU. GET YOUR MOVIE TICKET ONLINE NOW!
HOGWARTS EXPRESS
PHILIPS

A Common Mistake

- Advances in computer hardware (faster processor) will take care of the temporal requirements needed by a real-time system.
  - ✓ NO!
  - ✓ A Pentium 100 MHz with special care of real-time constrinats can be used to run a real-time application
  - ✓ However, a last generation pc (Pentium 4 2GHz) with a general purpose operating system (windows 2000, linux, etc.) can violate even a very simple temporal constraints of a real-time application.

- Rather than being fast, a real-time computing system should be *predictable*.
  - – low-level programming: aware of exact cycles for running a task
  - – theoretical proof of meeting all temporal constraints even in the worst case

Skills we have to learn as a real-time embedded system designer

- HW-related programming
  - full control of entire Hardware – from CPU to peripherals devices
  - Sometimes assembly programming for performance reason
  - In-depth understanding of operating system supports
- Dealing with Real-Time requirements
  - Task scheduling
  - Timing validation