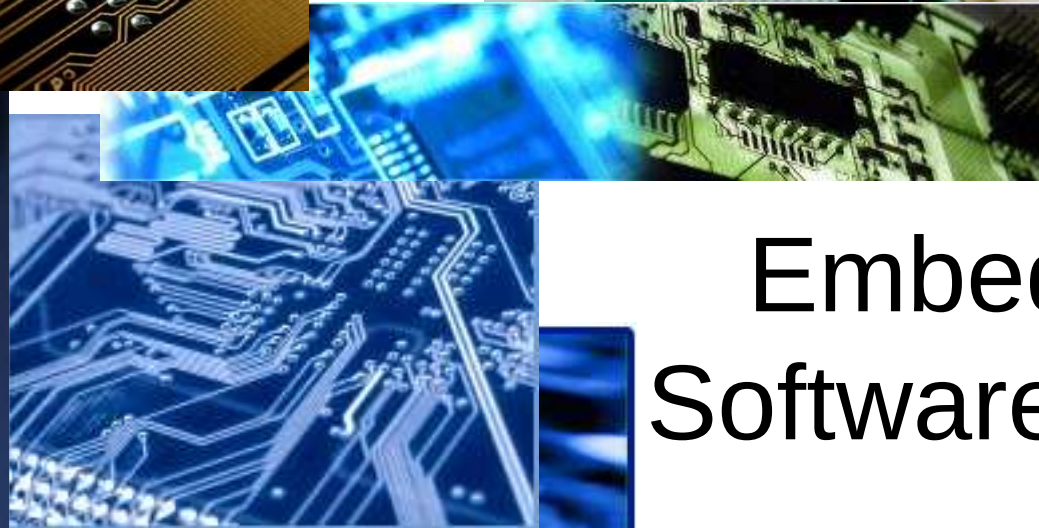
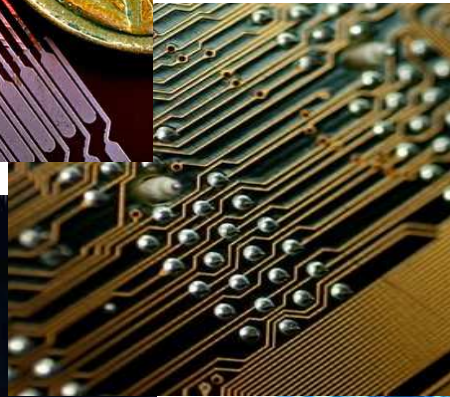
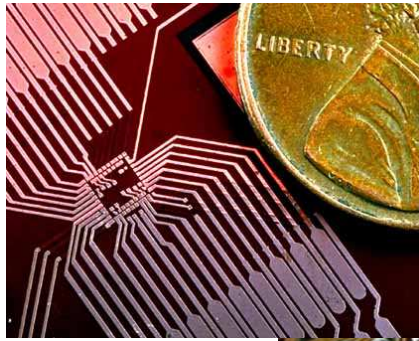
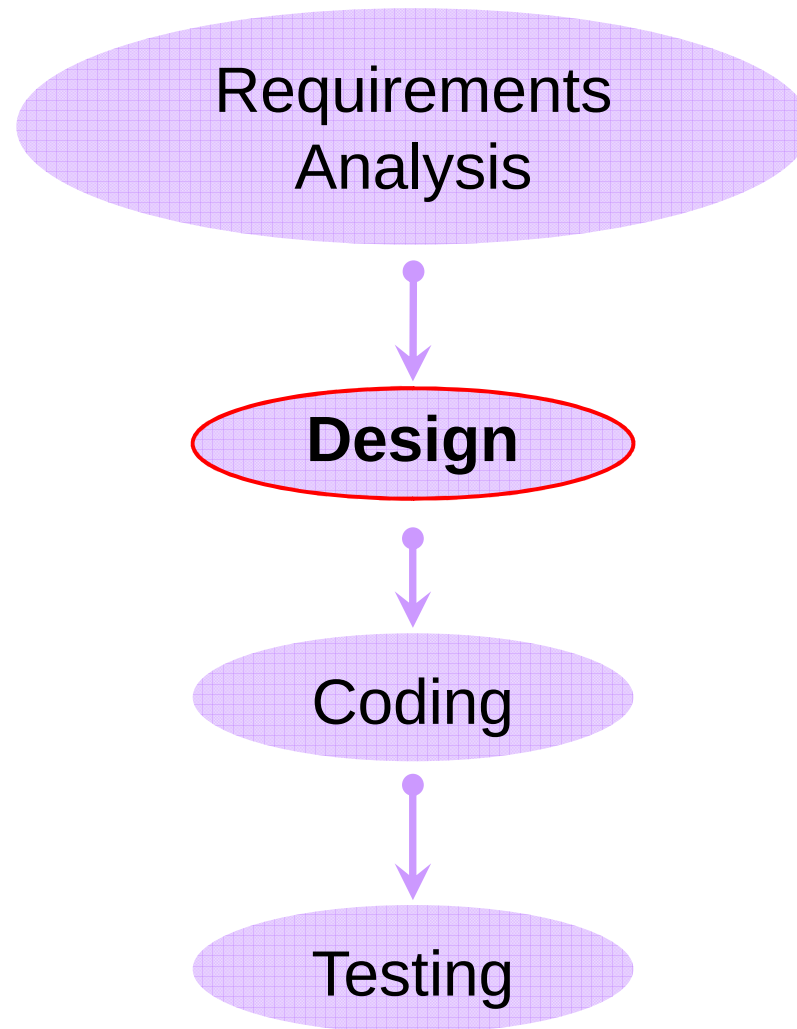




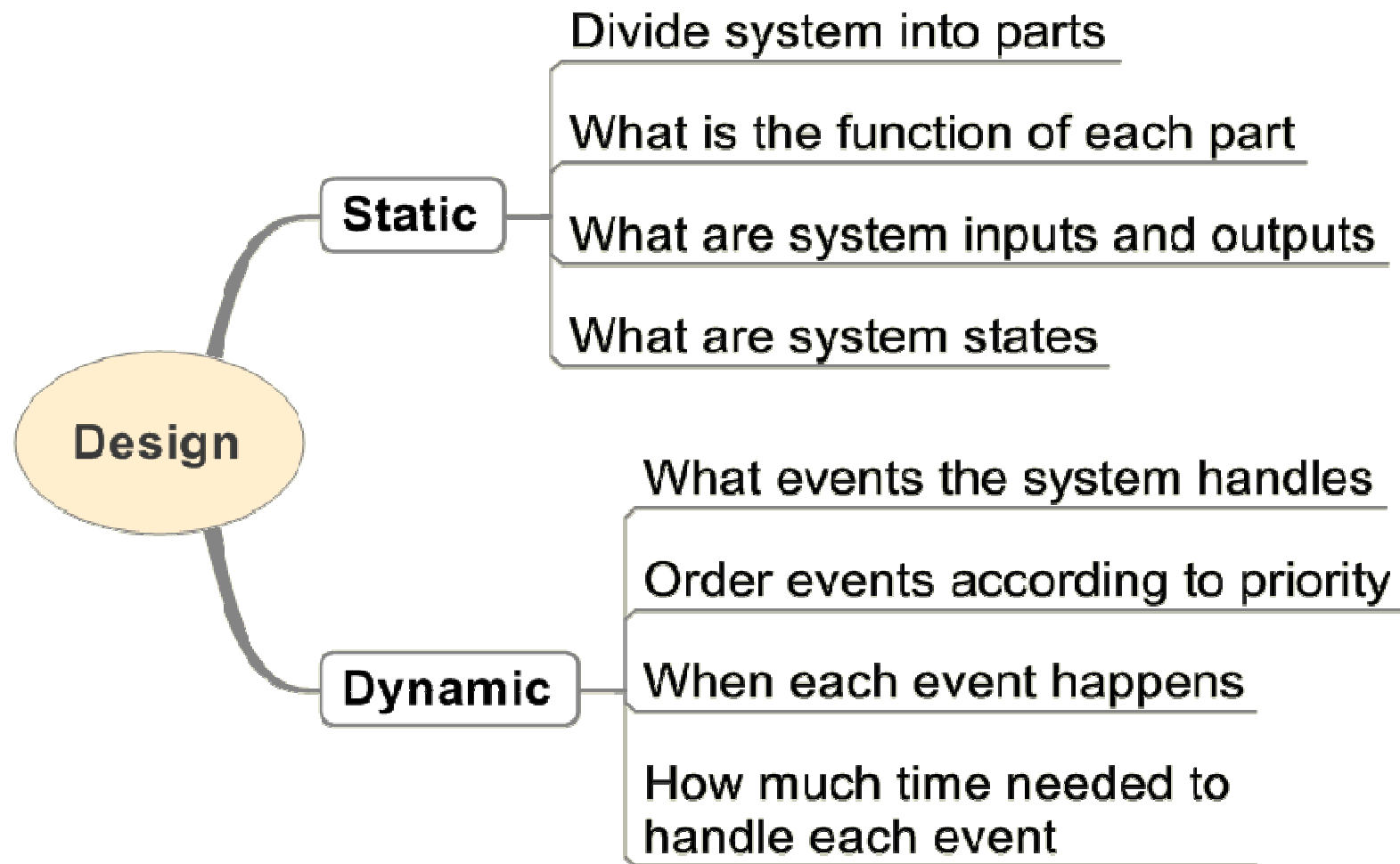
Embedded Software Cycle

Heba Sadek





Design



Dynamic Design

Dynamic

- ✓ Define events and their priorities
- ✓ Define when to handle each event
- ✓ Define time needed to handle each event

Real-time system

Actions are done on time (not fast)

Every action has a specific time to start and end

Embedded System

Functionality is continuously active as long as the switch is ON

Dynamic Design

Event

Something that happens in the system environment and the system should do something about it. Changes of the inputs that requires some action from the system

Example:

Temperature control system

Increase in temperature is an event

Decrease in temperature is an event

New wanted temperature from the user is an event

Dynamic Design

Task

- o Actions that handle an event
- o Since events are repeated all the time a task also is repeated as long as the system is switched ON
- o We can say that an embedded system is simply an infinite loop where functions are called to detect and handle events
- o Those functions are called **Tasks**

Example:

Code of a simple embedded system

```
main()
{
    //Initialization

    while(1)
    {
        Task_1();
        Task_2();
    }
}
```

Dynamic Design

- To complete system design you need to define all events the system should handle
- And define how they will be handled → Tasks
- Some events are more critical than others and should be handled first → **Priority**

How does the system detect an event so that it can be handled?

1. **Interrupt** → system is forced to deal with the event as soon as it happens
2. **Polling** → system should check periodically if the event has happened then deals with it (function called periodically)

Dynamic Design

```
InitTimer(100);          /*100ms*/
```

```
while(1)
{
    OutputLCD();
}
```

```
TimerISR()
{
    readSensor();
}
```

```
InitTimer(100);          /*100ms*/
```

```
while(1)
{
    TimerFlag = CheckTimerFlag();

    if(TimerFlag == 1)
    {
        readSensor();
    }

    OutputLCD();
}
```

```
while(1)  
{  
    if(TimerFlag == 1)  
    {  
        readSensor();  
    }  
    OutputLCD();  
}
```

```
timerISR()  
{  
    TimerFlag = 1;  
}
```

Dynamic Design

Example:

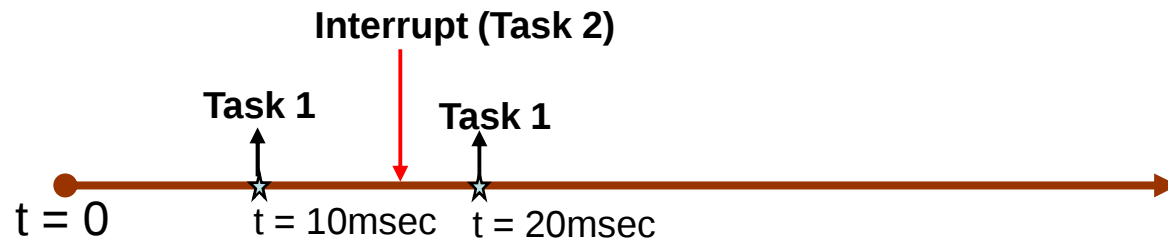
For the Tank Temperature controller
We can define the following tasks:

Task 1: check temperature every 10 msec

The function (task) that handles temp. change is called every 10 msec
(polling using a timer)

Task 2: an interrupt will be active if a user pressed a key then the ISR
will read value pressed on keypad

Timeline →



Dynamics Design

Traffic Light Controller

LAB 3

1. Define events handled by the system
2. Define tasks and their priority
3. Choose how to handles each event (interrupt or polling)
4. Draw system timeline