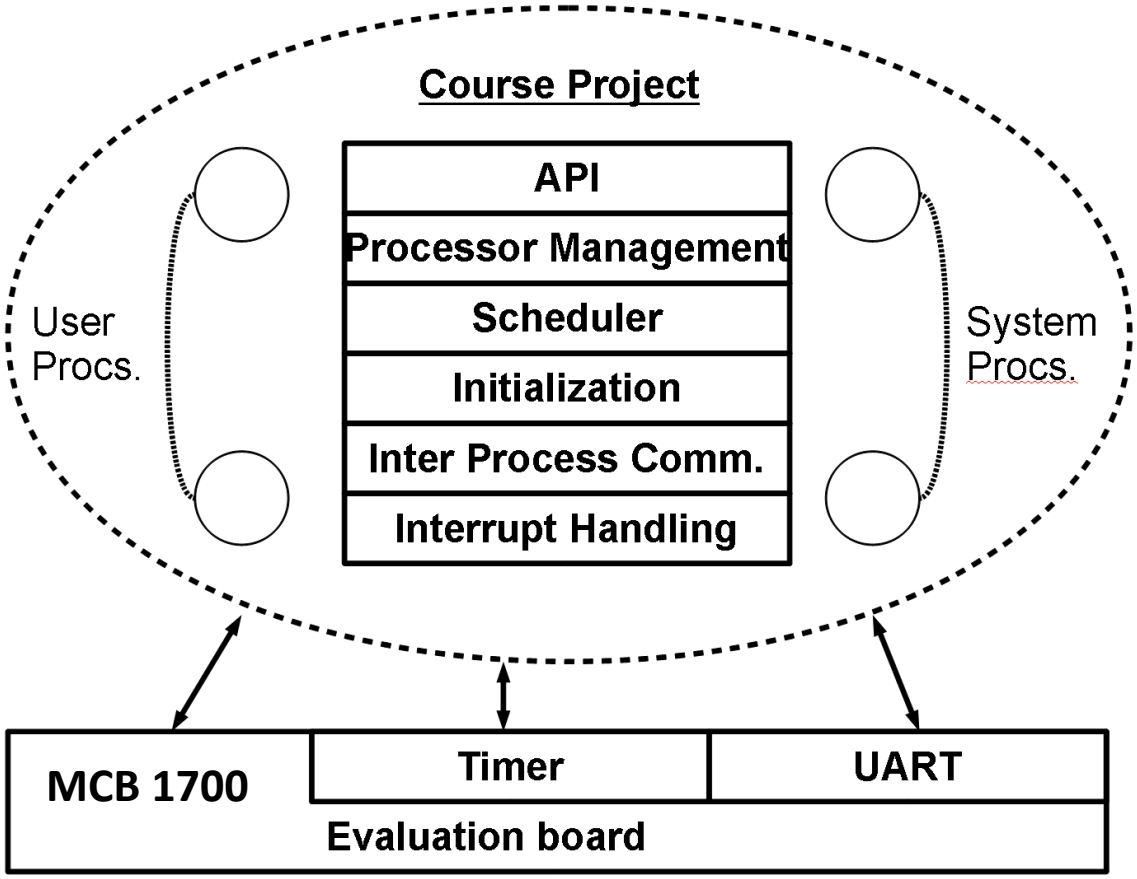
# SE350 – Project Overview and Memory Management

Nabil Drawil

# Functional Overview



# Deliverables

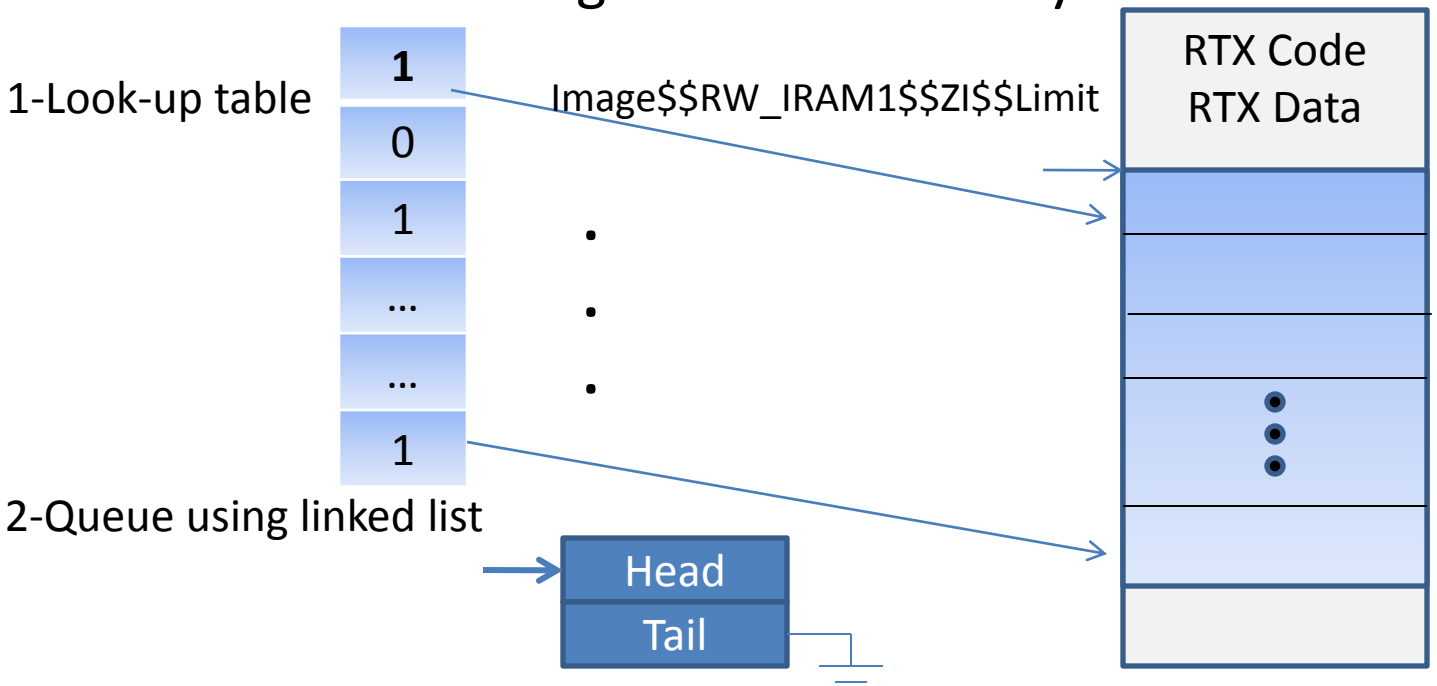
| Project Parts | Requirements | Submissions | Deadlines |
|---|---|---|---|
| RTX Project P1 | Memory management (data structure + APIs) Specified processes in the SPECs as well as few testing processes | Source code + Documentation | Jan. 31st |
| RTX Project P2 | Simplified version of the RTX | Source code + Documentation | Mar. 07th |
| RTX Project P3 | Final version of the TTX | Source code + Documentation | Mar. 25th |
| RTX Project P4 | Final project documents | Documentation | Apr. 04th |

# Memory Management

- In order to facilitate inter-process communications, messages are required
- Messages are basically memory blocks granted by the OS
- Memory blocks should be collected by the OS after they have been used
- The OS returns the released blocks to the free memory space

# Memory Management

- Free Memory Space
  - Where does it start?
  - Initialization
  - Data structure to organize the memory utilization



# Request Memory Block

- Pseudo code:

```
void * s_request_memory_block()
{
        search for free memory block in the pool;
        if (no memory block is available)
                return NULL;
        else
                update the data structure;
                return a pointer to the memory block;
}
```

# Release Memory Block

- Pseudo code

```
int s_release_memory_block(void* memory_block)
{
        if (memory block pointer is not valid)
                return ErrorCode;
        Add memory block to the pool;
        Update datastructure;
        returen SuccessCode;
}
```

# Questions?