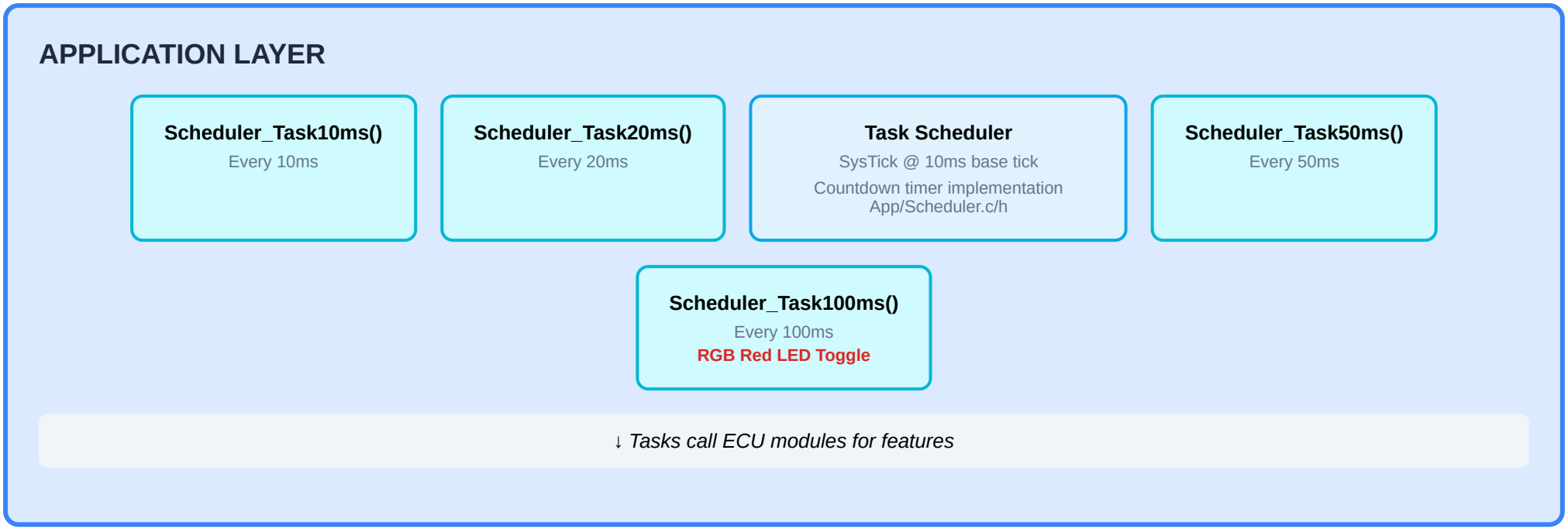
# TBSAR Block Diagram - Scheduler and Layers



## ECU LAYER (Feature Modules)

**Rgb**
RGB LED Control
P2.10, P1.2, P1.10

**Sensors**
ADC + Conversions
LM35, VPOT

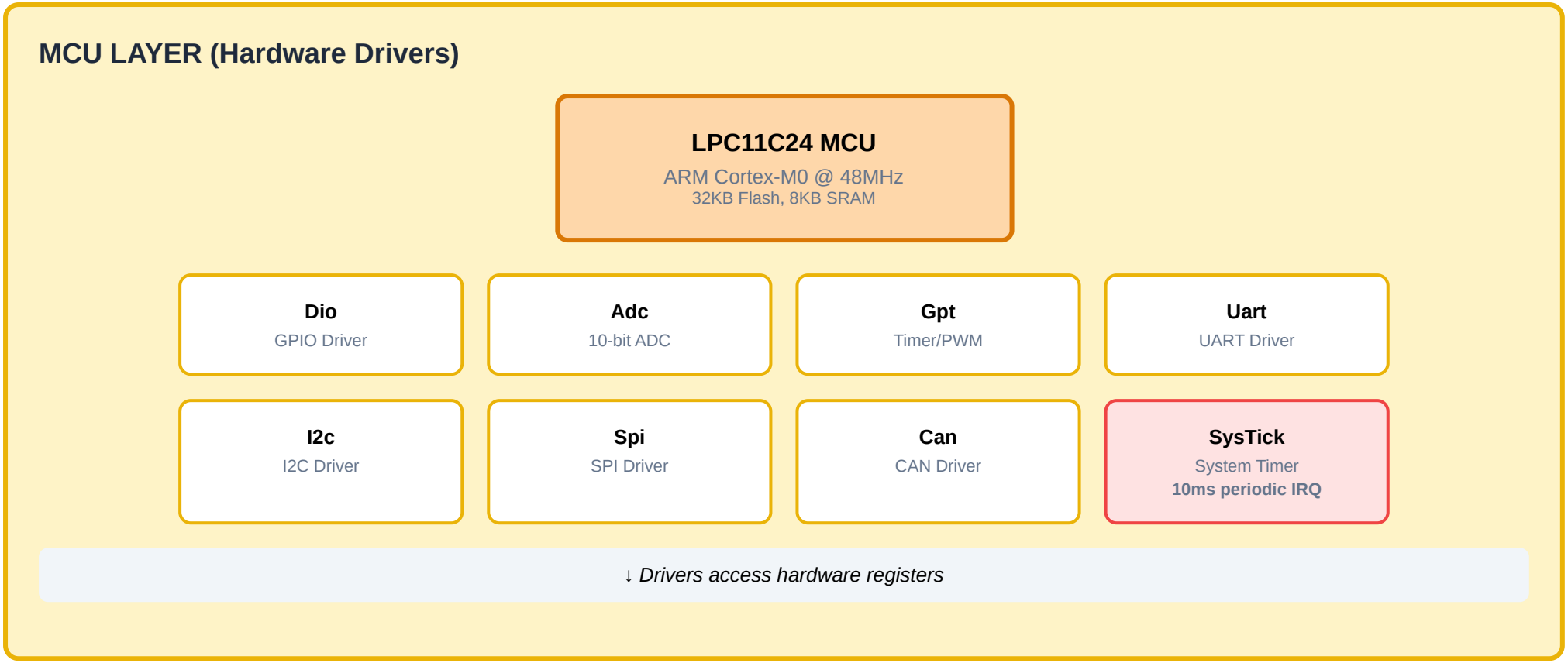
**Ukeys**
5 User Keys
P0.3, P1.4, P2.6-8

**Monitor**
UART Debug Printf
57600 baud

**Eeprom**
I2C Memory
400kHz

**Dac**
SPI DAC
1MHz

↓ *ECU modules call MCU drivers*


MCU LAYER (Hardware Drivers)
LPC11C24 MCU
ARM Cortex-M0 @ 48MHz
32KB Flash, 8KB SRAM
Dio
GPIO Driver
Adc
10-bit ADC
Gpt
Timer/PWM
Uart
UART Driver
I2c
I2C Driver
Spi
SPI Driver
Can
CAN Driver
SysTick
System Timer
10ms periodic IRQ
↓ Drivers access hardware registers

## Hardware Pin Mappings

| Module | MCU Driver | Pins | Details |
|---|---|---|---|
| **RGB LED** | Dio (GPIO) | P2.10 (Red), P1.2 (Green), P1.10 (Blue) | Active-high output |
| **User Keys** | Dio (GPIO) | P0.3, P1.4, P2.6, P2.7, P2.8 | 5 keys with pull-ups |
| **Sensors** | Adc | P0.11 (VPOT), P1.0 (LM35) | 10-bit ADC channels |
| **Monitor (UART)** | Uart | P1.6 (RXD), P1.7 (TXD) | 57600 baud, 8N1 |
| **EEPROM** | I2c | P0.4 (SCL), P0.5 (SDA) | 400kHz, 7-bit addressing |
| **DAC** | Spi | P2.11 (SCK), P0.9 (MOSI), P0.2 (CS) | 1MHz SPI clock |

## Current Application: Task Scheduler with RGB LED

### Scheduler Implementation:

- Base tick: 10ms (configured via SysTick_Config())
- Task periods: 10ms, 20ms, 50ms, 100ms
- Implementation: Countdown timers in SysTick_Handler()
- Files: App/Scheduler.c, App/Scheduler.h
- Current task: RGB Red LED toggle every 100ms (Scheduler_Task100ms)

### Build Information:

- Flash usage: 7668 bytes (23.6% of 32KB)
- RAM usage: 584 bytes (7.1% of 8KB)
- Toolchain: arm-none-eabi-gcc with -O2 optimization
- Programming: lpc21isp via UART @ 57600 baud
- MCU: NXP LPC11C24 (ARM Cortex-M0 @ 48MHz)

## Scheduler Execution Flow

### 1. SysTick Interrupt (every 10ms):

- Hardware timer generates periodic interrupt
- SysTick_Handler() in App/Scheduler.c is called
- Countdown timers are decremented

### 2. Task Execution:

- When a task counter reaches zero, corresponding task is called
- Task_10ms: counter = 1 (runs every 10ms)
- Task_20ms: counter = 2 (runs every 20ms)
- Task_50ms: counter = 5 (runs every 50ms)
- Task_100ms: counter = 10 (runs every 100ms, toggles RGB LED)

### 3. Module Access:

- Tasks call ECU modules (e.g., Rgb_Set(RGB_RED))
- ECU modules call MCU drivers (e.g., Dio pin control)
- Drivers write to hardware registers
- Hardware peripherals respond