# Esami Tecnologie dei Sistemi di Controllo

# Modalità di svolgimento telematico della prova scritta + tesina

**PREMESSA**

L'organizzazione delle prove a distanza implica difficoltà per entrambe le parti coinvolte, docenti e studenti. A questi ultimi è richiesta di questi tempi più che mai **maturità, responsabilità e massima collaborazione**. Infatti, tali prove a distanza richiedono necessariamente la suddivisione di un gruppo numeroso di partecipanti ad un appello in gruppi più piccoli, compatibili con l'accesso ad una sessione telematica tecnicamente gestibile (i.e. 10-12 studenti per gruppo). Considerando la complessità di preparare tali sessioni in modo compatibile con le esigenze di tutti gli studenti, **chi non abbia maturato una adeguata preparazione è caldamente invitato a non confermare l'iscrizione all'esame ed eventualmente cancellarla PRIMA della scadenza dei termini di iscrizione. Tali termini sono sempre fissati ad almeno 4 giorni prima dell'appello effettivo. Non saranno inoltre ammesse richieste via email di iscrizione tardiva.**

**NOTA TECNICA**

Le modalità descritte in questo documento sono l'adattamento al contesto specifico di questo insegnamento delle linee guida generali definite dall'Università di Ferrara con l'ausilio di tecnici informatici e docenti membri del Presidio Qualità, tratte da:

http://www.unife.it/it/covid19/docenti

vedi "Esami online - linee guida per esami scritti" con **versioni anche per gli studenti**.

e di ulteriori linee guida condivise personalmente tra docenti del Dipartimento di Ingegneria.

## Modalità di verifica dell'apprendimento

L'esame consiste nella presentazione di una tesina di approfondimento e in una prova con **8 domande a risposta multipla su sensori/trasduttori nella Parte A delle dispense.** Le domande ammettono UNA sola risposta giusta. Una risposta sbagliata corrisponde a una penalizzazione di 0,5 punti. Una risposta NON fornita corrisponde a 0 punti. La prova si ritiene superata se lo studente consegue una valutazione complessiva maggiore o uguale a 18. **Non** sono previsti punteggi minimi separati per le due parti dell'esame.

Il progetto svolto e documentato nella tesina deve essere **personale**, cioè sviluppato e presentato da ogni studente in autonomia.

**Prima di iniziare lo sviluppo di una tesina**, lo studente dovrà inviare una e-mail al docente per descrivere il proprio proposito di progetto, sia estratto tra i **suggerimenti elencati nell'appendice di questo documento** sia di propria inventiva. A seguito di tale contatto il docente valuterà l'attinenza come tesina per il corso e fornirà eventuali suggerimenti ulteriori.

Il progetto dovrà essere accompagnato da una breve relazione di alcune pagine, con la descrizione del lavoro svolto, il codice sviluppato, grafici o figure o foto dei risultati sperimentali e commenti personali.

**Dopo aver ricevuto tale relazione via e-mail dallo studente, il docente fisserà un incontro telematico personalizzato** per discutere dei contenuti e, possibilmente, per

visionare una dimostrazione pratica. **Tale presentazione dovrà precedere lo svolgimento della prova di teoria**.

Nella prova di teoria, svolta in corrispondenza delle date di appello pubblicate via https://studiare.unife.it, allo studente saranno concessi **25 minuti di sessione telematica per rispondere alle 8 domande.**

**NOTA**: le risposte fornite trascorsi i 25 minuti della sessione telematica NON verranno considerate corrette.

Durante la prova non è consentita la consultazione di testi, appunti o altro materiale. Non è inoltre ammesso l'utilizzo di auricolari o cuffie.

Qualsiasi inosservanza delle regole e raccomandazioni che seguono o problemi di tipo tecnico comporteranno l'esclusione immediata dalla prova e l'invalidazione della stessa.

## Svolgimento della prova di teoria

Ogni iscritto all'appello riceverà, almeno **18 ore** prima della prova, un invito a partecipare ad una conference call di Google Hangouts Meet. Chi non possa partecipare alla sessione cui è stato invitato dovrà comunicarlo tempestivamente al docente che provvederà all'inserimento in una conference call successiva (con un preavviso di almeno **12 ore**). In seconda convocazione **si dovrà essere presenti**. Chi, senza aver dato alcun avviso, non sarà presente alla conference call cui è stato invitato verrà classificato come ASSENTE.

**Prima di accedere** è necessario aver preparato adeguatamente la postazione sulla quale si intende svolgere la prova. In particolare è richiesto:

1) Di lasciare sulla scrivania esclusivamente il seguente materiale: documento di identità e tastiera/monitor del PC usato per la prova.
2) Di utilizzare una webcam esterna (e.g. USB) al fine di inquadrare: sé stessi, il tavolo o altra superficie d'appoggio su cui si scriverà ed il monitor citato prima. Chi non disponga di una webcam esterna dovrà eseguire l'accesso alla conference call **anche tramite smartphone** ed utilizzare questo come webcam esterna (l'utilizzo di Google Hangouts Meet dallo smartphone richiede l'installazione della relativa app). Si raccomanda di verificare accuratamente la configurazione prima di accedere alla prova, al fine di evitare inutili ritardi. La figura seguente riporta una situazione adeguata allo svolgimento della prova.

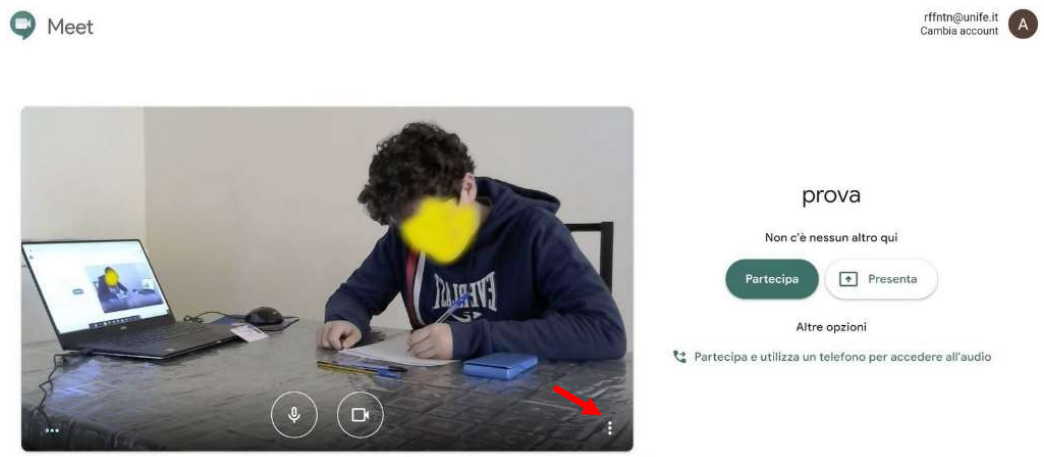


Figura 1

**Configurazione Audio/Video:** una volta effettuato l'accesso, per garantire la migliore qualità audio/video del collegamento, seguire le seguenti istruzioni a seconda della configurazione scelta.

*Configurazione 1: PC/Notebook + Webcam esterna connessa al PC/Notebook*

Sul **PC/Notebook** entrare nelle impostazioni della conference call cliccando sul simbolo ⋮ (indicato dalla freccia rossa in figura 1) e selezionando la voce *Impostazioni*. Nella finestra che si aprirà (figura 2), procedere come segue:

- *Sezione Audio*: Selezionare il microfono e gli altoparlanti integrati nel PC/Notebook.
- *Sezione Video*: Selezionare la webcam esterna.
- Confermare cliccando su *Fine*.

Cliccando sul pulsante "partecipa" in figura 1 si accede alla conference call: attivare il microfono, il video e gli altoparlanti del PC/Notebook.

*Configurazione 2: PC/Notebook + Smartphone come webcam esterna*

Sul **PC/Notebook (o tablet)** entrare nelle impostazioni della conference call cliccando sul simbolo ⋮ (indicato dalla freccia rossa in figura 1) e selezionando la voce *Impostazioni*. Nella finestra che si aprirà (figura 2), procedere come segue:

- *Sezione Audio*: Selezionare il microfono e gli altoparlanti integrati nel PC/Notebook.
- *Sezione Video*: Selezionare la webcam integrata nel PC/Notebook.
- Confermare cliccando su *Fine*.

Cliccando sul pulsante "partecipa" in figura 1 si accede alla conference call con il PC/Notebook: attivare il microfono e gli altoparlanti, disattivare invece il video.

Sullo **smartphone**: entrare nella conference call, quindi abbassare il volume dello smartphone (possibilmente disattivarlo), disattivare il microfono e attivare il video.

Si consiglia di mantenere lo smartphone connesso al caricabatteria durante la prova, onde evitare disconnessioni improvvise del collegamento a causa dell'esaurimento della batteria.

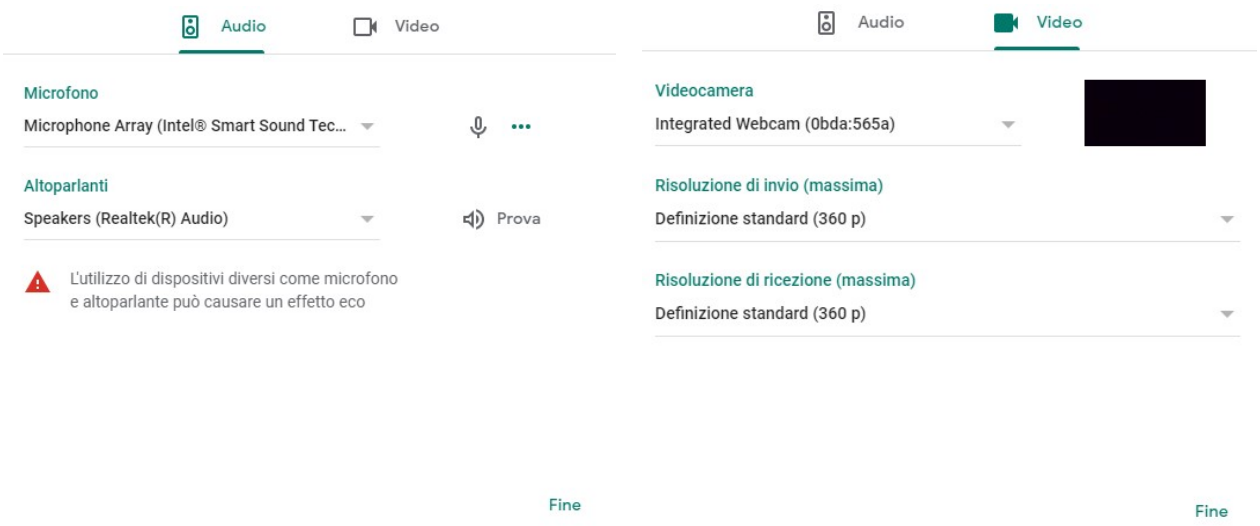


Figura 2

**La prova si articola in due fasi:**

1. Riconoscimento dello studente.
2. Soluzione delle domande a risposta multipla **tramite Google Form** (25 minuti).

**Riconoscimento dello studente**

Ogni studente sarà chiamato personalmente e dovrà mostrare alla webcam il documento di identità (con foto) ed il proprio viso.

**NOTE:**

Durante la fase successiva il docente potrà richiedere in ogni momento allo studente di visualizzare tramite webcam/smartphone tutto il suo ambiente circostante, allo scopo di verificare il corretto svolgimento della prova.

Durante la sessione il microfono dovrà essere tenuto acceso.

Non si potrà abbandonare la postazione, se non per gravi e giustificati motivi, durante la prova.

In caso di assenza prolungata della connessione di rete, il docente si riserva la possibilità di escludere lo studente dalla sessione, con l'opzione di inserirlo in una sessione successiva con il minimo preavviso compatibile con tempi tecnici.

Le domande al docente devono essere ridotte al minimo (disturbano gli altri studenti), la comprensione del testo è quindi parte integrante della prova.

**Soluzione delle domande a risposta multipla (25 minuti)**

I quiz a risposta multipla saranno erogati tramite piattaforma Google Form. Il link con il modulo dei quiz sarà inviato per email ad ogni studente all'inizio della sessione. Il tempo di svolgimento sarà vincolato dalla piattaforma stessa, perciò risposte tardive (i.e. dopo i 25 minuti concessi) NON saranno considerate corrette.

Durante lo svolgimento di questa parte ciascuno studente dovrà **condividere lo schermo intero** del proprio PC/notebook o tablet, dopodichè dovrà **impostare la finestra con il**

**Google Form in modalità pieno schermo** e mantenere tale modalità fino al termine del tempo concesso, **anche dopo aver terminato di rispondere**.

Operativamente:

1. Dalla sessione Google Meet, cliccare su "**Presenta ora**" in basso a destra
2. Cliccare su "**Il tuo schermo intero**"
3. Cliccare sull'icona del proprio schermo
4. Cliccare su "**Condividi**"

**Si vedano anche tali passaggi evidenziati nella Figura 3**

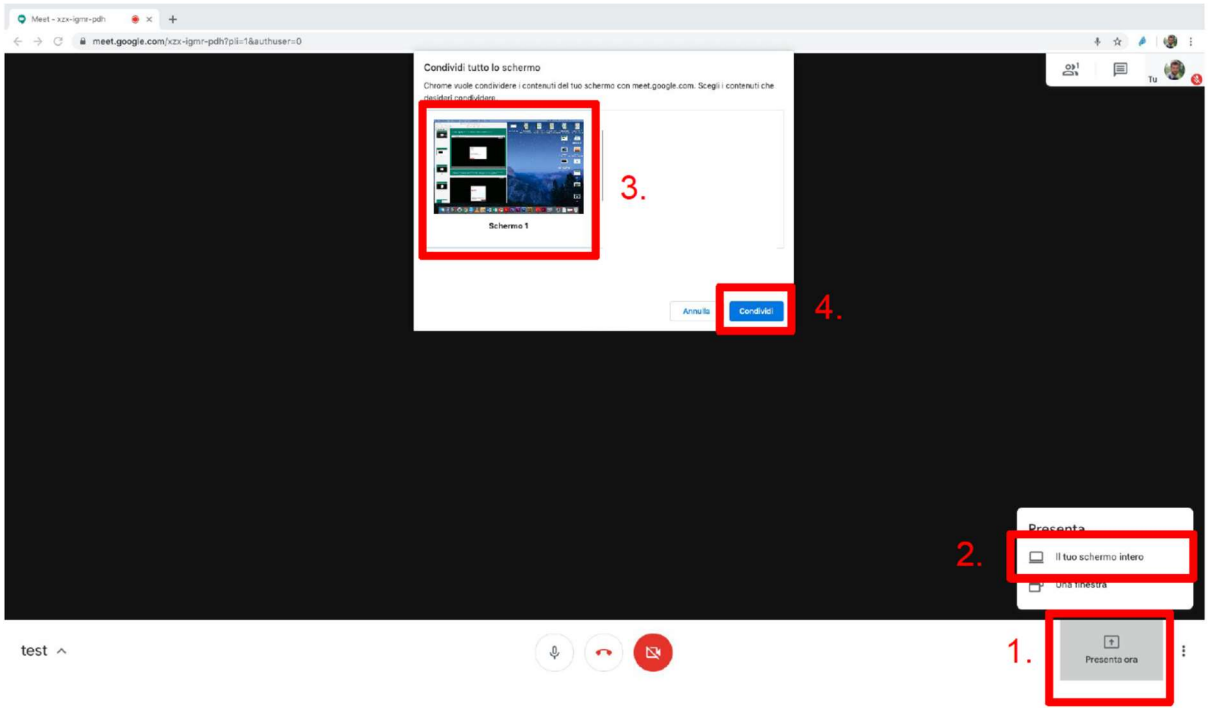


Figura 3

**Conclusione della prova**

**Alla scadenza del tempo assegnato** i risultati saranno raccolti automaticamente dal Google Form.

Chi finisca la prova anticipatamente o voglia ritirarsi dovrà comunque **aspettare lo scadere del tempo** previsto per la prova, segnalando al docente la propria intenzione tramite la Chat della sessione Google Meet.

**NOTA:** è possibile ripetere l'esame in appelli successivi, nel qual caso verrà considerato valido il voto conseguito nell'ultima prova CONSEGNATA.

Ferrara, 20/05/2020

Firma

(Marcello Bonfè)

**APPENDICE: suggerimenti per tesine di approfondimento**

## 1. Implementazione del controllo PID fixed-point per il motore BLDC

Strumenti utilizzabili:

- MPLAB X con compilatore XC16
- Schematico Proteus VSM per la simulazione dell'AN957 di Microchip
- Code Example CE019 (per dsPIC30F, adattabile a dsPIC33): http://ww1.microchip.com/downloads/en/DeviceDoc/CE019_PID.zip
- Documentazione ulteriore nella cartella "docs\dsp_lib" del percorso di installazione di XC16 (es. C:\Program files\Microchip\xc16\v\.50)

Il compilatore XC16 fornisce una libreria specifica per sfruttare pienamente il DSP engine dei dsPIC30/33, con funzioni implementate in assembler ma con prototipo (i.e. header file) per richiamare tali funzioni in linguaggio C. In particolare le funzioni per il controllo PID permettono di eseguire l'algoritmo di controllo in soli 33 cicli di istruzione, con uscita codificata in formato Q15 (a 16 bit), quindi nativamente scalata tra -1.0 e 0.9999 (considerati anche come valori di saturazione.

L'utilizzo pratico del PID "fractional" richiede quindi di scalare opportunamente l'errore di controllo (es. differenza tra velocità desiderata e misurata, in RPM) prima di passarlo come parametro nella chiamata del PID, in modo che sia compreso tra -1.0 e 0.9999 (con ipotesi ragionevoli sui valori massimi/minimi delle velocità misurabili..), poi scalare l'uscita del PID in modo tale che il valore -1.0 corrisponda al minimo della variabile di comando (es. 0% di duty cycle del comando PWM per il convertitore di potenza del motore) e 0.9999 al massimo (es. 100% del duty cycle).

## 2. Implementazione di un'applicazione (es. uno degli esercizi mostrati nel corso o una relativa variante) con il MPLAB Device Blocks per Simulink

Strumenti utilizzabili:

- MPLAB X con compilatore XC16
- Qualunque schematico Proteus con dsPIC33
- Matlab/Simulink con Matlab Coder, Simulink Coder e Embedded Coder
- MPLAB Device Blocks per Simulink, scaricabile tramite https://microchipdeveloper.com/simulink:start oppure https://github.com/LubinKerhuel/MPLAB-Device-Blocks-for-Simulink

Tramite i blocchi dell'MPLAB Device Blocks è possibile creare l'applicazione completa per un dsPIC30/33 con uno schema Simulink. Tale schema deve includere alcuni blocchi di preparazione del codice, cioè blocchi senza ingressi o uscite ma con un menu contestuale dedicato alla configurazione del processore (es. clock, configuration bits, ecc.), i blocchi di configurazione delle periferiche (es. ADC, McPWM ecc.), blocchi direzionabili DAGLI ingressi analogici/digitali e VERSO le uscite fisiche digitali/PWM del dsPIC e qualunque altro blocco di elaborazione matematica supportato da Simulink.

Ad esempio, sarebbe possibile realizzare un filtraggio di segnali analogici acquisiti tramite ADC del dsPIC oppure modulare i segnali PWM in modo da ricreare dei pattern sinusoidali (vedi https://www.eeeguide.com/sinusoidal-pulse-modulation/) oppure ancora ricreare il controllo closed-loop già descritto al punto 1, ma in modo interamente grafico.

Avendo a disposizione una scheda target supportata come Programmer/Debugger (es. Microstick II o un programmatore della famiglia PicKit) sarebbe possibile anche trasferire direttamente l'applicazione compilata usando solamente comandi di Matlab/Simulink. Nel caso di utilizzo di Proteus VSM bisognerebbe invece generare SOLO il codice sorgente (da Simulink: Apps → Embedded Coder → Generate Code invece di Build) per poi importarlo in un progetto MPLAB X creato ad-hoc.

**NOTA**: allargando il contesto ad altri microcontrollori/DSP/DSC si trovano molti altri "blockset" o toolbox specifici per la generazione automatica di codice C/C++ ottimizzato per questi target. Si potrebbe quindi sviluppare anche una tesina relativa all'uso di tali strumenti, per chi abbia già o voglia procurarsi una propria scheda con hardware differente da dsPIC33. Si veda https://it.mathworks.com/discovery/simulink-embedded-hardware.html per una panoramica completa dell'hardware supportato.

### 3. Migrazione di un'applicazione Arduino per un target dsPIC33

Strumenti utilizzabili:

- MPLAB X con compilatore XC16
- Qualunque schematico Proteus con dsPIC33
- Esempi di librerie "Arduino-like" per altri microcontrollori/DSC Microchip:
  https://circuitcellar.com/cc-blog/execute-open-source-arduino-code-in-a-pic-microcontroller-using-the-mplab-ide/
  http://kibacorp.com/free-downloads
  https://chipkit.net/wiki/index.php?title=MPLABX_Importer

Gli esempi citati forniscono strumenti per convertire "sketch" per Arduino in programmi compilabile in ambiente MPLAB X. Purtroppo, si riferiscono entrambi all'utilizzo di PIC32. Tuttavia, la conversione di un insieme base di tali librerie (es. funzionalità elementari per configurazione pin, lettura/scrittura di I/O digitali, lettura di segnali analogici) in modo che siano compatibili ALMENO con il dsPIC33FJ12GP202 o MC202, visti nelle videolezioni di esercitazione con il simulatore Proteus VSM sarebbe un interessante approfondimento di vari contesti della programmazione embedded.

### 4. Sviluppo di un'applicazione multi-tasking con FreeRTOS per dsPIC33

Strumenti utilizzabili:

- MPLAB X con compilatore XC16
- Qualunque schematico Proteus con dsPIC33
- Porting del sistema operativo per applicazioni embedded FreeRTOS: https://www.freertos.org/portpic24_dspic.html
- MPLAB X Add-on FreeRTOS Viewer (per visualizzare le caratteristiche dei task programmati)

FreeRTOS è un insieme di librerie con funzioni di sistema operativo compatibili con le limitate capacità di memoria e calcolo dei sistemi embedded, per i quali però fornisce primitive relativamente semplici per la definizione di:

- Task periodici
- Sincronizzazione fra i Task (es. semafori/mutex)
- Gestione di comunicazioni inter-task (code, stream buffers, ecc.)

Un primo obiettivo potrebbe essere quello di testare esempi forniti con il codice sorgente del Porting per dsPIC/PIC24 adattandoli alle caratteristiche dei dsPIC33 inseriti negli schematici Proteus VSM mostrati nelle videolezioni del corso. Un obiettivo più ambizioso, che richiede però anche l'uso di hardware reperito autonomamente dallo studente, potrebbe essere quello di sviluppare una piccola applicazione IoT grazie all'integrazione di servizi forniti da Amazon con FreeRTOS: https://github.com/MicrochipTech/amazon-freertos

## 5. Programmazione di schede Arduino tramite MPLAB X

Strumenti utilizzabili:

- MPLAB X con compilatore XC8 o XC32 (a seconda dell'Arduino usato)
- Scheda Arduino reperita autonomamente dallo studente
- Esempi di procedura proposta per altri target (i.e. schede Arduino-like ma con processori Microchip PIC32): https://chipkit.net/programming-chipkit-boards-in-mplabx/
- Esempi di procedura per Arduino ma che richiedono hardware aggiuntivo: https://microchipdeveloper.com/mplabx:avr-debugwire

Da quando Microchip ha acquisito Atmel, quest'ultima produttrice dei microcontrollori AVR/ARM su cui sono basate le schede Arduino, è possibile programmare tali microcontrollori con MPLAB X, i relativi compilatori ed alcuni Programmer/Debugger prodotti dopo la fusione tra le due aziende (es. PicKit4). Tuttavia, com'è noto le schede Arduino sono pre-programmate con un bootloader che permette un più semplice ed economico (NO hardware aggiuntivo) trasferimento del codice da Arduino IDE.

Sviluppare e testare una procedura per programmare schede Arduino tramite il bootloader nativo, ma utilizzando MPLAB X come ambiente di sviluppo è quindi un interessante approfondimento su varie tecnologie per sistemi embedded.

## 6. Cenni generali su come impostare un proprio progetto con attinenza al corso

Proseguendo le considerazioni su schede Arduino, essendo nota la loro diffusione tra hobbisti e, presumibilmente, anche tra studenti di Ingegneria, si forniscono nel seguito alcune linee guida sull'uso di tali schede in modo che una qualunque relativa applicazione risulti compatibile con il corso in oggetto:

- **Configurazione tramite SFR**: uno degli obiettivi del corso è l'apprendimento delle metodologie di configurazione ed uso di periferiche integrate in microcontrollori/DSC, tramite la comprensione delle impostazioni associate ai relativi SFR. Qualunque sia l'applicazione sviluppata con una scheda Arduino, se questa è programmata configurando l'hardware direttamente tramite SFR anziché sfruttando le librerie di Arduino IDE, risulterà certamente istruttiva e professionalizzante tanto quanto gli esempi proposti nell'ambito delle lezioni di Tecnologie dei Sistemi di Controllo. A puro titolo di esempio, sono forniti nel seguito alcuni link a pagine che trattano l'argomento della programmazione *avanzata* di schede Arduino:

https://www.arduino.cc/en/Reference/PortManipulation
http://www.gammon.com.au/adc
http://www.gammon.com.au/interrupts
http://nicecircuits.com/playing-with-analog-to-digital-converter-on-arduino-due/
http://www.atwillys.de/content/cc/using-custom-ide-and-system-library-on-arduino-due-sam3x8e/?lang=en
http://forum.arduino.cc/index.php?topic=140205.0

- **Uso di strumenti di debug**: Arduino IDE non permette l'esecuzione di codice in modalità debug, sia per la filosofia stessa di trasferimento del codice sulla scheda target tramite bootloader che per la inerente semplicità (o anche, povertà funzionale..) dell'IDE stesso. Esistono però dei modi, sebbene perlopiù basati su hardware aggiuntivo, per effettuare un debug approfondito di un'applicazione Arduino, operazioni che costituiscono anch'esse un importante competenza nel settore dei sistemi embedded. A titolo di esempio:
  https://sites.google.com/site/wayneholder/debugwire
  https://starter-kit.nettigo.eu/2015/debug-sketch-on-arduino-zero-pro-with-gdb-and-openocd/
  https://devblogs.microsoft.com/iotdev/debug-your-arduino-code-with-visual-studio-code/
  https://www.codeproject.com/Articles/5150391/Creating-and-Debugging-Arduino-Programs-in-Visual
  https://www.codeproject.com/Articles/5160447/Creating-and-Debugging-Arduino-Programs-in-Visua-2
  https://hackaday.io/project/162302-debugging-arduino-uno-in-vscode

Infine, oltre alle schede Arduino anche i Single-Board Computer come **Raspberry Pi** hanno conquistato uno spazio rilevante nel panorama dei sistemi embedded per l'hobbistica. Raspberry Pi è però un target molto differente da quello di interesse principale per il corso di Tecnologie dei Sistemi di Controllo per i seguenti motivi:

- Non integra nativamente periferiche specifiche per il controllo (es. ADC o PWM ad altra frequenza per controllo motori o convertitori di potenza).
- L'eventuale acquisizione di sensori o il comando di motori richiedono normalmente hardware aggiuntivo la cui interfaccia avviene tramite bus di comunicazione I2C/SPI, che rappresentano tecnologie significativamente diverse da quelle presentate nel corso citato.
- Essendo a tutti gli effetti dei Computer (Single-Board) con sistema operativo Linux-based, la programmazione è di fatto analoga a quella dei contesti non embedded, pertanto anch'essa significativamente diversa da quella presentata nel corso.

Tuttavia, un argomento interessante e potenzialmente attinente alle Tecnologie per Sistemi di Controllo è l'installazione su Raspberry di un sistema operativo con capacità Real-Time. A tale proposito, si forniscono nel seguito alcuni link a puro titolo di esempio:

https://www.socallinuxexpo.org/sites/default/files/presentations/Steven_Doran_SCALE_13x.pdf

https://www.stevebate.net/chibios-rpi/GettingStarted.html

https://easychair.org/publications/open/VPzR

https://www.get-edi.io/Real-Time-Linux-on-the-Raspberry-Pi/

https://lemariva.com/blog/2019/09/raspberry-pi-4b-preempt-rt-kernel-419y-performance-test