UNIVERSITÀ POLITECNICA DELLE MARCHE
FACOLTÀ DI INGEGNERIA
Corso di Laurea Magistrale in Ingegneria Elettronica

Progettazione dell'interfaccia hardware per la prototipazione di chip virtuali

Design for hardware interface needed for virtual chip prototyping

Tesi di Laurea di:
Martina Greco

Relatrice:
Dott.ssa **Laura Falaschetti**

Correlatore:
Ing. **Matteo Silvestrini**

Anno Accademico 2022/2023

Indice

Capitolo 1

Introduzione

Al giorno d'oggi l'innovazione può favorire una più elevata produttività e ciò consente di erogare nuovi servizi e beni volti al miglioramento della vita degli utenti che usufruiscono dei progressi digitali. Anno dopo anno, infatti, le aziende e le persone che vi lavorano trovano modi nuovi e creativi per migliorare i loro prodotti.
Naturalmente questo non si limita solo ai prodotti finiti, ma si estende ad ogni altro prodotto della catena di produzione.
Il settore dei semiconduttori ha un ciclo di sviluppo dei prodotti lungo e costoso soprattutto quando si tratta di applicazioni automobilistiche, che hanno requisiti aggiuntivi di sicurezza funzionale. Prima che un prodotto venga immesso sul mercato sono necessari numerosi processi di progettazione. Ogni processo di progettazione richiede molto tempo a causa delle continue revisioni e, successivamente, del relativo arrivo dei prototipi.
Infineon Technologies è la più grande azienda tedesca produttrice di semiconduttori per l'elettronica di potenza: produce componenti per l'alimentazione di azionamenti elettrici, motori, elettrodomestici, strumenti per l'elaborazione dati, sistemi elettromedicali e apparati d'illuminazione. Offre soluzioni a semiconduttore e di sistema per i settori automotive, industriale e chip-card, come circuiti integrati di gestione dell'alimentazione, convertitori DC-DC, LDO e circuiti integrati per il controllo dei motori ed è particolarmente interessata a realizzare prototipi pre-silicio dei propri sistemi analogici a segnale misto.

Tradizionalmente, i chip analogici a segnale misto dovevano essere prototipati producendo diverse revisioni di ASIC (Application-Specific Integrated Circuit). Questo non solo ha un costo molto elevato ma ha anche diversi effetti ambientali legati all'energia e all'uso di risorse su un progetto che probabilmente non verrà mai utilizzato al di fuori del laboratorio. Da qui nasce, appunto, la necessità di una procedura di prototipazione accelerata per mantenere il proprio vantaggio innovativo. Quando si parla di prototipazione vengono subito in mente le FPGA (Field Programmable Gate Array), poiché lavorano su logica programmabile e riproducono fedelmente il funzionamento di un ASIC senza richiedere la creazione del progetto. Questo aiuta ad accelerare il processo di sviluppo eliminando qualsiasi bug che potrebbe non essere visibile nella verifica logica e ciò è già ampiamente utilizzato nell'industria.

La sfida di questa tesi consiste nel progettare una piattaforma hardware pre-silicio che sia in grado di replicare il comportamento di un generico regolatore LDO integrato in un **PMIC** (Power Management Integrated Circuits) per poter accelerare i processi di verifica in laboratorio: il PMIC è un dispositivo che aiuta a gestire i requisiti di alimentazione del sistema host. Questo tipo di circuito integrato gestisce il flusso di alimentazione da varie fonti. Tali dispositivi stanno diventando sempre meno costosi e più piccoli grazie ai progressi tecnologici.
Inoltre, il principale campo di utilizzo dei PMIC è quello automotive e la sua efficienza di gestione è strettamente legata alla sua struttura.
Per poter interfacciare la board progettata con la logica integrata nell'FPGA, c'è stata l'esigenza di aggiungere elementi circuitali esterni all'LDO per implementare alcuni controlli necessari per chiudere il loop PCB-FPGA, garantendo così l'accelerazione dei test di laboratorio.
Pertanto, esploreremo se e come un approccio di piattaforma virtuale possa essere implementato utilizzando modelli analogici sia fisici che virtuali, in modo da accelerare lo sviluppo del prodotto e migliorare il time-to-market.

La struttura con cui si procederà alla presentazione della tesi è illustrata di seguito. Nel capitolo 2 verrà presentato lo stato dell'arte ed i relativi articoli scientifici che hanno già trattato il tema del virtual chip prototyping. Nel capitolo 3 verranno descritti tutti gli strumenti hardware e software utilizzati per tale progetto di tesi. Il capitolo 4 si concentrerà, invece, sul setup dell'FPGA e sulla comunicazione tra host ed FPGA. Il capitolo 5 analizzerà i passi per la realizzazione della PCB progettata. Il capitolo 6 conclude la tesi, facendo punto sul grado di innovazione introdotto nel lavoro proposto attraverso test svolti in laboratorio.

Capitolo 2

Stato dell'arte

Sono stati fatti molti studi in merito alla prototipazione di chip virtuali.
Con il continuo progresso della tecnologia del silicio e l'elevata richiesta di funzionalità per l'elettronica di consumo, i software embedded stanno diventando sempre più complessi e iniziano a superare i contenuti hardware nel costo di sviluppo.
La prototipazione consiste nel creare modelli virtuali 3D dei prodotti e di effettuare test su tali modelli, tramite appositi software, per simulare il comportamento dei prodotti nel mondo reale.

2.1 Prototipazione virtuale di sistemi embedded su chip

In [1] vengono illustrate le attuali pratiche industriali di prototipazione virtuale del silicio, ne identifica le sfide e invita a trovare nuovi modi per affrontarle.
Il continuo progresso delle tecnologie di fabbricazione ha portato la complessità dei System-On-Chip (SOC) ai massimi storici, con miliardi di transistor su un singolo chip. Allo stesso modo, il successo dell'elettronica di consumo ricca di potenzialità ha portato ad una crescente domanda di funzionalità differenziate per il software.
Pertanto, oggi i SOC non sono più dei circuiti integrati specifici per un'applicazione (ASIC) che eseguono una o poche funzioni, ma piuttosto piattaforme che consentono

e addirittura favoriscono lo sviluppo di applicazioni. Questa tendenza sta spingendo i produttori di sistemi e SOC a fornire un software completo insieme al software di sistema come il firmware, i driver e il sistema operativo (OS).
Lo sviluppo del software risulta essere la parte più grande e in più rapida crescita della progettazione totale di un SOC pertanto il costo proporzionale dello sviluppo di software, anche con le migliori pratiche di riutilizzo di esso, è aumentato vertiginosamente, come mostrato nella figura sottostante.

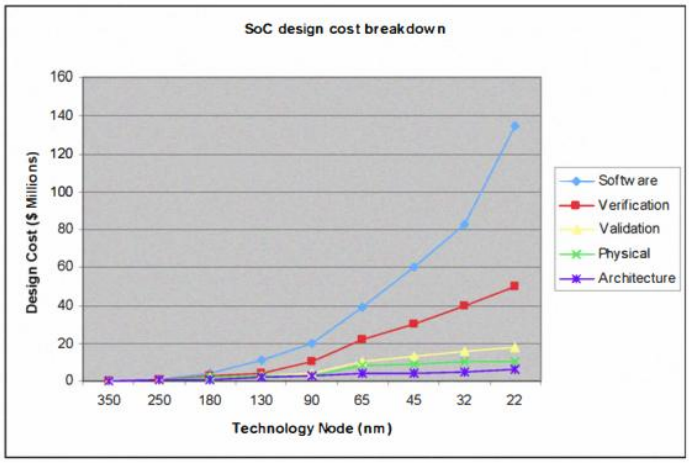


Figura 2.1: Ripartizione dei costi di progettazione del SOC per diversi nodi tecnologici

L'industria dell'automazione della progettazione elettronica ha da tempo individuato che il tradizionale approccio alla progettazione di SOC incentrato sull'hardware, in cui l'hardware viene prima progettato e poi consegnato al team software per l'inizio dello sviluppo, non funziona più a causa dell'aumento dei tempi di sviluppo del software. È diventato presto chiaro che un quadro di co-sviluppo hardware/software è la chiave del successo di un progetto, in quanto consente al team software di iniziare a prototipare e sviluppare prima che la scheda hardware sia tornata dagli impianti di fabbricazione. Tale attività di sviluppo è tipicamente chiamata prototipazione virtuale. Uno strumento chiave utilizzato per la prototipazione virtuale è la piattaforma virtuale (VP) o un insieme di modelli funzionali dell'hardware del SOC, come processori, periferiche e bus, in forma di software. Il bus e le periferiche sono tipicamente modellati usando un linguaggio di alto livello come ANSI C o SystemC con particolare

attenzione alle interfacce visibili del software, come i registri, il protocollo del bus e le funzionalità delle periferiche.

Utilizzando VP come piattaforma hardware proxy, gli ingegneri software possono iniziare il loro lavoro mentre gli ingegneri hardware eseguono la progettazione a livello di blocco o l'integrazione a livello di chip e scheda.

Un tipico flusso di progettazione tradizionale di un SOC inizia con lo sviluppo dell'hardware che consiste nella progettazione di algoritmi, progettazione dell'architettura, sviluppo dei componenti di proprietà intellettuale (IP), integrazione del chip, verifica.

Una volta che l'intero ciclo hardware è terminato e la scheda è arrivata, il team software inizia l'insieme di attività di raccolta ed elaborazione dei dati e lo sviluppo del software sulla scheda.

Data la pressione del time-to-market, il tradizionale flusso di progettazione è stato migliorato con la prototipazione virtuale per consentire lo sviluppo precoce del software prima dell'arrivo della scheda di valutazione hardware con i chip prodotti.

Esistono diverse tecnologie di prototipazione virtuale che vanno dalle simulazioni RTL ad alta precisione di ciclo, all'emulazione in circuito (ICE), alle piattaforme virtuali. Tutti e tre gli approcci hanno i loro vantaggi e svantaggi.

Le **simulazioni RTL** modellano con precisione comportamenti dell'hardware fino al livello di trasferimento dei registri, adatti alla verifica e al profiling dell'hardware.

D'altra parte, **ICE** emula l'intero SOC sintetizzando ed eseguendo l'RTL del SOC su Field Programmable Gate Array (FPGA). Questo fornisce tempo di esecuzione ragionevolmente veloce per lo sviluppo del software.

Il **VP** di un sistema completo è un modello software funzionale dell'hardware del SOC: di solito consiste in modelli ISS (Instruction Set Simulator) e bus/periferiche puramente funzionali.

I modelli sono implementati concentrandosi sulla funzionalità piuttosto che sui dettagli dell'implementazione per ottenere una maggiore velocità di simulazione.

Un tipico VP è in grado di simulare l'avvio di Linux in decine di secondi, il che lo rende sufficientemente veloce per lo sviluppo di software complessi. Poiché richiede

solo la specifica funzionale dell'hardware, il VP può essere costruito dopo la fase di algoritmo del ciclo di progettazione: questo permette allo sviluppo del software di iniziare molto prima nella tempistica del progetto.
Una piattaforma virtuale (VP) è un insieme di modelli funzionali di software e hardware che insieme formano un SOC singolo o multicore.
I prototipi virtuali vengono eseguiti su un host, spesso un PC generico, e sfruttano l'ISS per eseguire software embedded di produzione non modificato, comprese le applicazioni, OS, firmware e driver.
La velocità di simulazione dipende dall'hardware. Tuttavia, per lo sviluppo del software, la precisione del ciclo non è un fattore importante perchè spesso il software:

- Non si preoccupa dell'implementazione dell'hardware ma delle sue funzionalità;
- Non ha bisogno di conoscere l'esatto comportamento temporale;
- Interagisce direttamente solo con le interfacce hardware (ad esempio, registri e bus di memoria).

Pertanto, i modelli hardware in VP tendono a implementare solo le funzionalità e le interfacce visibili dal software, risultando modelli molto più semplici.
I modelli sono spesso implementati utilizzando linguaggi di alto livello come SystemC o C/C++ puro. Le comunicazioni tra modelli si ottengono chiamando le funzioni di interfaccia del modello di destinazione, ponendo l'accento sulle funzionalità del trasferimento di dati (quali dati vengono trasferiti a e da) invece che sulla sua effettiva implementazione.
Il VP offre una serie di vantaggi rispetto ad altre metodologie di simulazione basate sull'hardware:

1. Disponibilità anticipata del software: la disponibilità di VP consente al team software di iniziare a fare test nelle prime fasi del progetto. Il flusso di progettazione del SOC con l'uso di VP può consentire alla progettazione del software di iniziare subito dopo la fase di algoritmo hardware;

2. Migliore visibilità degli stati del software e dell'hardware: poiché il VP è completamente implementato in modelli software controllati da un kernel di simulazione, non è necessario osservare gli stati del software e dell'hardware, come registri e linee di interruzione, in qualsiasi momento;

3. Velocità di simulazione: grazie all'elevata astrazione dei dettagli dell'hardware, un tipico VP può raggiungere una velocità di distruibuzione dei dati pari a decine di milioni di istruzioni al secondo (MIPS: million instructions per second).

Le elevate prestazioni di simulazione consentono agli sviluppatori di software di avviare stack di software complessi come Linux embedded in decine di secondi, rendendolo adatto per il complesso sviluppo di software embedded di oggi.
Nell'industria esiste un'ampia varietà di applicazioni di VP e la maggior parte di esse sono progettate internamente per soddisfare le esigenze di sviluppo dei SOC dell'azienda.
Sebbene il VP offra vantaggi significativi al team software, rimangono diverse sfide tra cui:

- Riduzione velocità: istruzioni come la conversione del set di istruzioni di destinazione in quello di set di istruzioni dell'host, tendono ad essere più lente dell'esecuzione del software nativo sull'host e ciò porta ad un rallentamento dei processi;

- Usabilità: diversi strumenti di sviluppo software sono di solito disponibili gratuitamente per i computer da tavolo ma spesso non sono disponibili per i processori embedded, o almeno richiedono un notevole sforzo di porting infatti la maggior parte dei processori embedded è dotata solo del minimo indispensabile di strumenti necessari per la compilazione;

- Dipendenza dall'hardware: il software embedded deve quasi sempre interagire con le caratteristiche dell'hardware. Questo crea un'altra catena di dipendenze anche in presenza di prototipi virtuali tra software applicativo, software operativo, software di prototipi virtuali, driver di dispositivo e firmware con modelli

hardware. La costruzione e la convalida di una catena completa di software, introduce una lunga latenza di sviluppo e questo costringe gli sviluppatori di applicazioni ad attendere l'arrivo del software di sistema.

Queste sfide sono difficili da affrontare, tuttavia esistono delle soluzioni: una soluzione è che la dipendenza dall'hardware spesso non deriva dalla dipendenza del set di istruzioni ma piuttosto da dettagli dell'organizzazione del computer, come la mappa degli indirizzi delle periferiche.
Un'altra soluzione è rappresentata dall'aumento della virtualizzazione nell'industria dei computer in generale e, di conseguenza, dalla disponibilità del set di istruzioni per la virtualizzazione in processori di base.
Infatti, per supportare la virtualizzazione, ovvero la possibilità di avere più macchine virtuali sulla stessa CPU fisica, sia Intel che AMD hanno introdotto un miglioramento dei loro set di istruzioni. Spesso queste funzionalità consentono il monitoraggio hardware delle istruzioni di accesso alla memoria per determinati indirizzi di interesse. Grazie a queste soluzioni, aumenta così la possibilità di avere più piattaforme virtuali.
In conclusione, in [1] si sostiene che la prototipazione virtuale fornisca un valore aggiuntivo per lo sviluppo dei system-on-chip. Tuttavia, le piattaforme virtuali, in quanto forma più promettente di metodologia di prototipazione virtuale, devono affrontare una serie di sfide, che richiedono ulteriori ricerche in quest'area.
Sicuramente i produttori di SOC continueranno a interessarsi a questo settore, in quanto il loro successo dipende non solo dalla competizione per la riduzione dei costi e dall'ingresso precoce nel mercato ma sempre di più da una competizione di applicazioni.

2.2 Sistema di emulazione di chip basato su FPGA

La prototipazione su FPGA è diventata una delle principali metodologie di verifica per la progettazione di hardware, lo sviluppo di test e la co-progettazione del software. Nel caso di sviluppo dei test, l'FPGA funge da DUT (Design Under Test) virtuale e i modelli di test vengono applicati da apparecchiature di test automatiche (ATE). In questo modo si verifica non solo il chip con i circuiti di progettazione per il test (DFT) e il programma di test, ma anche l'intera configurazione di test che comprende il DUT virtuale, la scheda di carico e le interconnessioni con l'ATE. Sebbene la piattaforma basata su FPGA sia ampiamente utilizzata per la verifica del programma di test dei circuiti integrati digitali, questa tecnica non viene utilizzata per i circuiti analogici e a segnale misto (AMS) a causa della difficoltà di implementare i circuiti AMS nelle FPGA.

In [2] viene illustrato lo sviluppo di una piattaforma di emulazione di test di circuiti AMS basati su FPGA. La metodologia proposta sfrutta le tecniche di modellazione a virgola fissa e di implementazione DSP facilitate dai più recenti FPGA per modellare i circuiti AMS. Il flusso di test proposto per i circuiti AMS aiuterà gli ingegneri addetti ai test ad iniziare il loro piano di test in concomitanza con gli ingegneri addetti alla progettazione ed a convalidarli molto prima del primo silicio.

La prototipazione su FPGA è diventata una delle principali metodologie di verifica per la progettazione hardware, lo sviluppo di test e la co-progettazione del software nell'area dei VLSI (Very Large Scale Integration è una denominazione generica che indica un'elevata integrazione di transistor all'interno di un singolo chip) digitali.

Nel caso del flusso di verifica VLSI tradizionale, lo sviluppo del piano di test (che comprende la programmazione dei test, la progettazione della scheda di carico, la programmazione dell'ATE, ecc.) inizia verso la fine del ciclo di progettazione (ad esempio, dopo il backend).

Il circuito può essere prototipato utilizzando un FPGA che funge da DUT (Design Under Test) virtuale e posizionato sulla scheda di carico al posto del silicio originale. In questo modo si verifica non solo il chip con il circuito di progettazione per il test ed

il programma di test, ma anche l'intera configurazione di test che comprende la scheda di carico, le interconnessioni con l'ATE, ecc. Tuttavia non è possibile verificare tutti i parametri di prova, in particolare quelli relativi alla frequenza, attraverso questo concetto di DUT virtuale in quanto la frequenza degli ASIC ad alta velocità può essere superiore a quella dell'FPGA.

Sebbene la piattaforma basata su FPGA sia ampiamente utilizzata per la verifica di programmi di test dei circuiti digitali integrati, questa tecnica non viene utilizzata per i circuiti analogici e a segnale misto (AMS) a causa della difficoltà di implementare i circuiti AMS nelle FPGA.

In questo lavoro, viene presentato un nuovo approccio per la progettazione di un sistema di emulazione di chip per circuiti analogici utilizzando FPGA.

La Fig. 2.2 mostra un flusso di test tradizionale per i circuiti AMS. In questo caso, la pianificazione dei test può essere avviata solo nelle fasi finali del flusso di progettazione, vale a dire durante la verifica di backend o di sign off.

Per la pianificazione dei test, è necessario un modello di simulazione del circuito AMS che può essere ottenuto solo dopo la progettazione schematica dei blocchi analogici e la progettazione RTL (o a livello di gate) dei blocchi digitali.

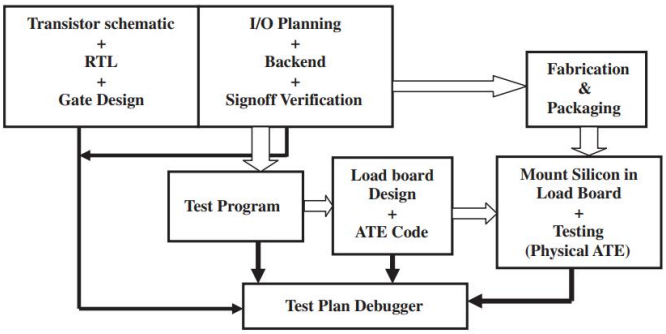


Figura 2.2: Traditional test flow

I problemi del flusso di test tradizionale sono i seguenti:

- La convalida del test program e del progetto della scheda di carico può essere fatta solo dopo aver progettato il test setup (comprendente la scheda di carico,

DUT fabbricato e montato sulla scheda di carico e collegato ad un ATE) e testato fisicamente il silicio;

- La pianificazione dei test inizia verso la fine della progettazione del front end e la validazione avviene solo dopo la fabbricazione del silicio.

Un flusso di test ideale è quello in cui il processo di progettazione e quello di pianificazione dei test procedono di pari passo, con un'interazione reciproca per una migliore ottimizzazione del piano di prova e per la testabilità del circuito di progettazione. La figura sottostante mostra questo flusso per i circuiti AMS, noto con il nome di "hand-in-hand test" e largamente utilizzato nei progetti digitali.

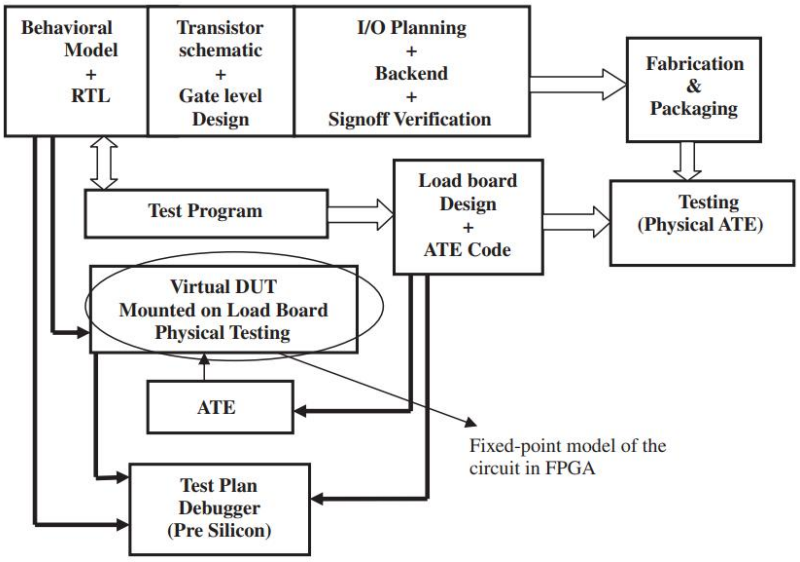


Figura 2.3: Hand-in-hand test planning and design flow

La ragione principale per cui questo flusso è realizzabile nel contesto digitale è la disponibilità di framework per la modellazione RTL di alto livello, ossia Verilog, VHDL, ecc.

Essi forniscono un modello di simulazione nelle fasi iniziali della progettazione per gli ingegneri di test in modo da poter iniziare la pianificazione dei test (test programming).

Inoltre, i progetti digitali possono essere facilmente prototipati in FPGA e trattati come DUT virtuali.

Per rendere possibile un tale flusso per i circuiti AMS, sono necessari due requisiti:

1. Modelli comportamentali di simulazione di alto livello (BM, behavioral models) per i circuiti AMS;

2. Un ambiente di prototipazione/emulazione in FPGA per i blocchi BM di alto livello per trattarli come DUT virtuali.

Il primo requisito facilita la modellazione dei blocchi a livello comportamentale che può servire come modello di simulazione senza la necessità di schemi a livello di transistor. Ciò consente di pianificare gli hand-in-hand tests per i circuiti AMS perchè i BM (analoghi ai modelli RTL nel caso dei circuiti digitali) possono essere sviluppati più rapidamente dei modelli schematici a transistor e possono essere resi disponibili nelle prime fasi del flusso di progettazione.
Per quanto riguarda il secondo requisito discusso in precedenza, i modelli a virgola fissa dei blocchi analogici devono essere sviluppati a partire dalle loro BM ed emulati in tempo reale su una FPGA ad alta velocità.
L'approccio per la verifica di circuiti integrati a segnale misto che utilizza la generazione di test basata su BM genera automaticamente i segnali di test (cioè il test program) per i circuiti AMS ed il processo può essere avviato nelle prime fasi del flusso di progettazione. Poiché l'approccio si basa solo sulla simulazione comportamentale piuttosto che sull'emulazione, non può essere utilizzato per generare un DUT virtuale per verificare i progetti delle schede di carico, l'impostazione dell'ATE, ecc.
Un metodo innovativo per eseguire simulazioni di circuiti simili a SPICE su una FPGA è presentato in [3], dove gli autori hanno simulato un filtro RC ed un transistor NMOS su una FPGA. Il lavoro si concentra principalmente sullo sviluppo di un ambiente di emulazione su FPGA per i modelli SPICE: questi non sono modelli di alto livello come i BM pertanto è difficile ottenere i modelli SPICE durante le prime fasi del flusso di progettazione e ciò porta ad un rallentamento per l'inizio della pianificazione dei test.
In letteratura sono stati riportati numerosi lavori in cui l'algoritmo di controllo di un convertitore buck DC-DC viene implementato in una piattaforma hardware basata su

FPGA. L'approccio per il controllo digitale completo utilizzando i circuiti FPGA per un convertitore buck DC-DC in tensione sfrutta principalmente la programmabilità dell'FPGA per implementare il controllo digitale senza componenti passivi esterni, mentre lo stadio di potenza è implementato in forma discreta all'esterno dell'FPGA. Oltre al controllo in modalità di tensione, sono stati riportati diversi approcci per l'architettura di controllo digitale per migliorare le prestazioni dei convertitori DC-DC. Queste tecniche di controllo sono state verificate utilizzando una piattaforma basata su FPGA o su DSP.
Il motivo della popolarità dell'FPGA è che fornisce un'architettura riconfigurabile, capacità di memoria e facile implementazione di molte funzioni DSP ad alta intensità di calcolo.

- **Approccio proposto per l'emulazione basato su Behavioral Models (BM)**

In questa sezione viene descritto l'approccio proposto per l'emulazione di circuiti AMS su FPGA.
L'approccio segue un flusso di progettazione top-down dell'FPGA; tale schema richiede due tipologie di strumenti di progettazione: il primo per lo sviluppo/analisi del BM e il secondo per la sintesi hardware e l'implementazione dei BMs. La progettazione dei BM può essere effettuata utilizzando Verilog-A, MATLAB-Simulink, Xilinx-System Generator (modellazione a punto fisso), ecc. La sintesi e l'implementazione dell'hardware su FPGA possono essere eseguite utilizzando Xilinx ISE, Aultera Quartus, ecc.
La prima fase di questo flusso consiste nello sviluppo del BM di ciascun modulo analogico del progetto target, sulla base delle specifiche fornite dai progettisti. Questa modellazione viene eseguita ad un alto livello di astrazione in un ambiente in virgola mobile utilizzando strumenti come MATLAB-Simulink o Verilog-A. Prima di passare alla fase successiva, è necessario assicurarsi che il modello in virgola mobile derivato corrisponda alle specifiche.

Il modello in virgola mobile consiste fondamentalmente in una serie di equazioni e funzioni di trasferimento che rappresentano il comportamento in tempo reale di ciascun modulo del progetto di destinazione. Di solito, per rappresentare i BM in virgola mobile si utilizzano i blocchi di Cadence Verilog-A o MATLAB-Simulink. La simulazione comportamentale viene eseguita per verificare se quello che si è ottenuto corrisponde a tutte le specifiche del progetto. Il BM in virgola mobile così ottenuto viene poi convertito in un modello in virgola fissa. Il modello a virgola fissa è rappresentato da una serie di equazioni, ottenute discretizzando il modello in virgola mobile con una frequenza di campionamento adeguata. La frequenza di campionamento è uno dei fattori principali che determinano l'accuratezza del modello a virgola fissa: la frequenza di campionamento f_s deve soddisfare il criterio di Nyquist $f_s \geq 2f_m$, dove f_m è la frequenza massima di funzionamento del progetto di destinazione.

Il limite superiore di f_s è vincolato dalla frequenza di conversione dei dati della scheda ADC-DAC da utilizzare per l'emulazione. Deve inoltre essere inferiore alla frequenza di clock massima alla quale il progetto sintetizzato può funzionare nell'FPGA.

I tipi di dati interni del modello a virgola fissa vengono convertiti in una rappresentazione bit-true che viene utilizzata nell'implementazione hardware. Il modello a virgola fissa con valori di coefficienti quantizzati e larghezza di bit dei dati limitata è simulato per verificare le sue prestazioni rispetto ai BM in virgola mobile.

Possono verificarsi problemi di overflow e saturazione che potrebbero portare a disallineamenti.

Quando il modello è stato convertito con successo in blocchi a virgola fissa, esso è realizzabile in FPGA. La logica di controllo e gli elementi per i quali non esistono macro hardware, vengono sintetizzati e tutti i pezzi vengono combinati in un'unica netlist RTL completamente realizzabile. È necessario sviluppare un wrapper HDL attorno alla netlist generata del progetto di destinazione per poter stabilire fondamentalmente un'interfaccia tra il mondo esterno e il progetto di destinazione. Il wrapper HDL contiene le informazioni sulla generazione del clock di campionamento dal clock di sistema dell'FPGA, sulla generazione del clock e sull'interfaccia ADC-DAC con il circuito. Gli strumenti Xilinx ISE tools vengono utilizzati per sintetizzare, posi-

zionare/instradare ed implementare la netlist RTL insieme al wrapper HDL in una FPGA.
I risultati di questo processo sono un bit-stream (file di programmazione FPGA) ed una netlist strutturale EDIF (Electronic Design Interchange Format) dell'hardware. Queste netlist possono essere simulate con gli stimoli di prova desiderati e verificare la corrispondenza con le specifiche.
Inoltre, per verificare le prestazioni della realizzazione basata su FPGA, il flusso di bit viene scaricato e testato fisicamente con i segnali di prova di un ATE (o di un'apparecchiatura di prova).
Una volta che la risposta dell'FPGA corrisponde alle specifiche (o simulazioni basate su BM), l'FPGA funge da un DUT virtuale.
In conclusione, la convalida del programma di test e il debug molto prima del silicio è un compito impegnativo nel caso di test analogici e a segnali misti.
La piattaforma di emulazione basata su FPGA mira principalmente a **ridurre la pressione del tempo dei test engineer**, fornendo una soluzione per avviare il piano di collaudo all'inizio del flusso di progettazione.

Gli studi effettuati e citati in [**?**] e [2] riguardano la prototipazione virtuale di chip e si focalizzano maggiormente sulla parte digitale, non considerando il lato hardware e la necessità di avere una PCB che si interfacci con l'FPGA e che replichi in laboratorio il comportamento del modulo del chip che si vuole andare a simulare.
Per tale motivo, la presente tesi si pone l'obiettivo di realizzare una board che si interfacci con l'FPGA e che permetta, in tal caso, di replicare il comportamento di un regolatore lineare di tensione.

Capitolo 3

Strumentazione software e hardware

Per la realizzazione di questo progetto sono stati utilizzati diversi strumenti hardware e software presenti nel laboratorio dell'azienda Infineon Technologies.

3.1 Ambiente di sviluppo LabVIEW

LabVIEW [4] è un ambiente di sviluppo integrato per il linguaggio di programmazione visuale di National Instruments (NI). Tale linguaggio grafico viene chiamato "Linguaggio G" in quanto si distingue dai linguaggi tradizionali per la sua sintassi grafica e non scritta. Per il lavoro di tesi qui esposto, la versione di LabVIEW utilizzata è la 2019.
Un programma in LabVIEW è denominato VI (Virtual Instrument), esso non esiste sotto forma di testo ma come un file binario che può essere aperto e compilato solo da LabVIEW.
La definizione di strutture dati ed algoritmi avviene tramite icone e altri oggetti grafici, ognuno dei quali incapsula funzioni diverse, uniti da linee di collegamento (wire) in modo da formare una sorta di diagramma di flusso.

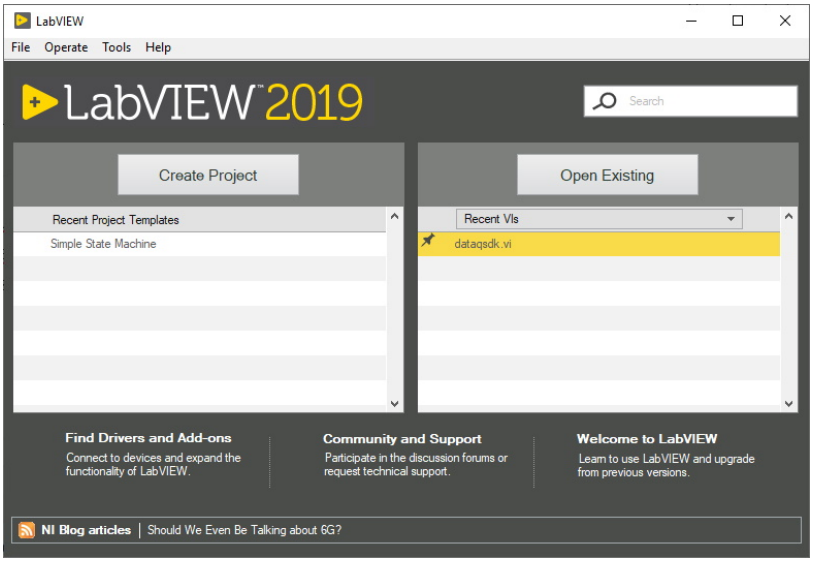
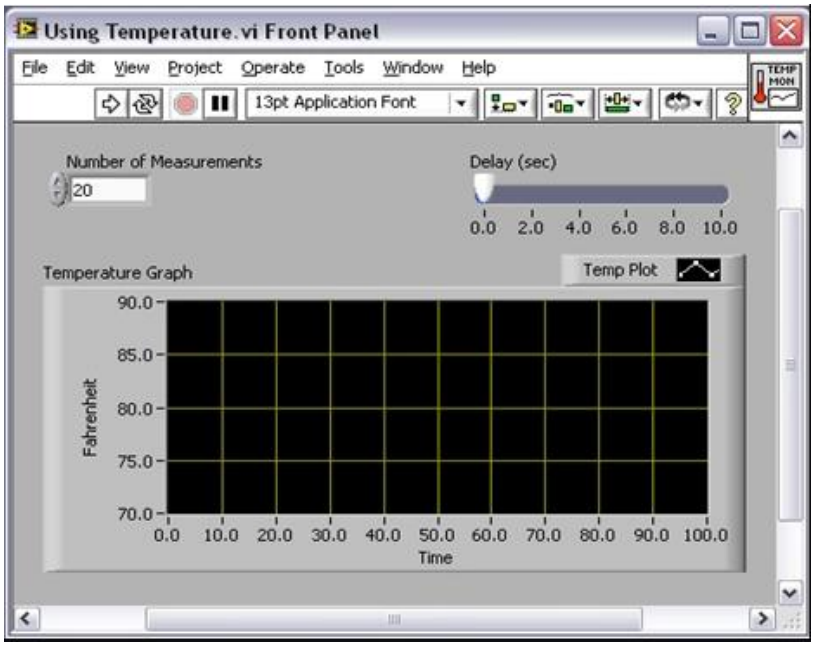


Figura 3.1: Ambiente di sviluppo LabVIEW 2019

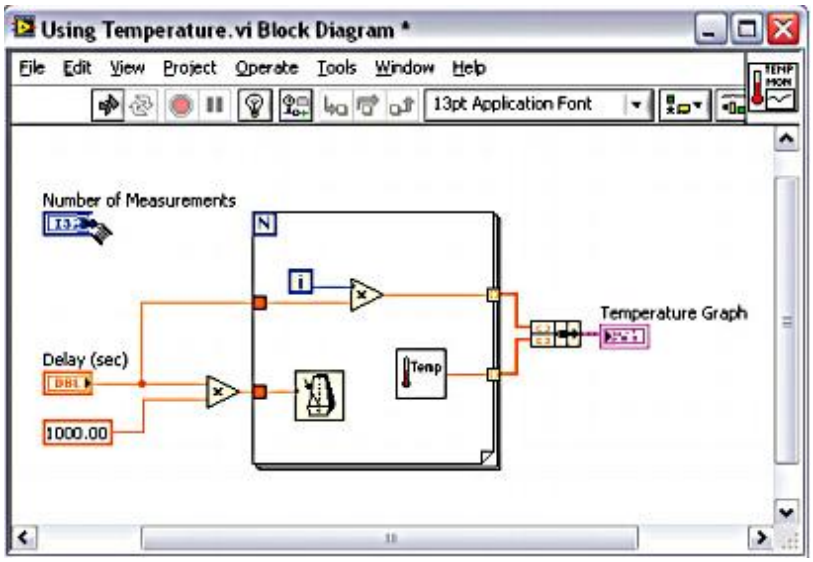
In particolare, un programma LabVIEW è composto da tre parti principali:

- il **pannello frontale**: è l'interfaccia utente del VI. Si realizza con controlli e indicatori che costituiscono, rispettivamente, i terminali interattivi d'ingresso e d'uscita.



I controlli possono essere matrici, manopole, potenziometri, pulsanti, quadranti e molti altri; simulano i dispositivi d'ingresso degli strumenti e forniscono dati allo schema a blocchi del VI. Gli indicatori possono essere grafici, tabelle, LED, termometri e molti altri; simulano i dispositivi d'uscita degli strumenti e visualizzano i dati che lo schema a blocchi acquisisce o genera;

- lo **schema a blocchi**: è il diagramma di flusso che rappresenta il codice sorgente, in formato grafico.



Gli oggetti del pannello frontale appaiono come terminali di ingresso o uscita nello schema a blocchi, mentre gli oggetti dello schema a blocchi comprendono: terminali, funzioni, costanti, strutture, chiamate ad altri VI, fili di collegamento, commenti testuali;

- il **riquadro connettori**: ogni VI può essere a sua volta utilizzato come subVI (o sottoVI) e comparire all'interno dello schema a blocchi di altri VI come una qualsiasi funzione e come tale può avere ingressi e uscite a cui collegare le linee di flusso. Il riquadro connettori serve pertanto a definire qual è l'aspetto del VI quando appare come subVI in uno schema a blocchi di un VI più ampio: che facciata ha l'icona, ma soprattutto come e dove vanno collegate le linee per permettere il passaggio dei dati.

In questo progetto, è stata utile l'estensione **LabVIEW FPGA Module** in quanto consente di sviluppare ed eseguire il debug della logica hardware personalizzata in modo che sia possibile compilare e distribuire su hardware FPGA NI.
LabVIEW FPGA è un software add-on per LabVIEW utilizzabile per progettare in modo più efficiente ed efficace i sistemi basati su FPGA mediante un ambiente di sviluppo altamente integrato, librerie IP, un simulatore ad alta fedeltà e funzionalità di debug. È possibile sia creare VI FPGA, che utilizzano l'accesso diretto agli I/O e la logica LabVIEW definita dall'utente, sia personalizzare l'hardware per applicazioni come le comunicazioni di protocollo digitale, la simulazione HIL (Hardware-In-The-Loop) e la prototipazione rapida di controllori. Dato che il modulo FPGA LabVIEW include routine di elaborazione del segnale, è possibile anche integrare il codice di HDL (Hardware Description Language) e IP di terze parti.

3.2 Chassis NI PXIe-1082

PXI (PCI eXtensions for Instrumentation) è una piattaforma basata su PC utilizzata per applicazioni di misurazione ed automazione quali test di produzione, monitoraggio di macchine e test industriali.

Figura 3.2: Chassis PXIe-1082

Tale piattaforma comunica con la scheda CompactPCI installata nel PC grazie ad un bus PCI (Peripheral Component Interconnect) ad alte prestazioni, con la possibilità di

avere a disposizione bus di sincronizzazione dedicati e funzionalità software avanzate. Lo chassis NI PXIe-1082 [5] a **otto** slot è dotato di un backplane ad alta larghezza di banda per soddisfare un'ampia varietà di applicazioni di test e misura ad alte prestazioni.
Accetta i moduli PXI Express in ogni slot e supporta moduli PXI standard ibridi compatibili fino a quattro slot. Lo chassis è caratterizzato da un intervallo di temperatura operativa esteso per applicazioni che necessitano di un sistema di raffreddamento.
In particolare, la NI PXIe-1082 contiene:

- il modulo **PXIe-8381**: modulo che gestisce la comunicazione tra PC e chassis PXI. Gestisce, inoltre, il bus di comunicazione tra i vari moduli installati;

Figura 3.3: Modulo PXIe-8381

- il modulo **PXI 7841R**: modulo che contiene la FPGA Xilinx *Virtex-5*, è in grado di offrire la possibilità di comunicare con altri dispositivi, grazie ai connettori IO, e di sviluppare diversi circuiti hardware di misurazione utilizzando l'ambiente di sviluppo grafico LabVIEW e LabVIEW FPGA.

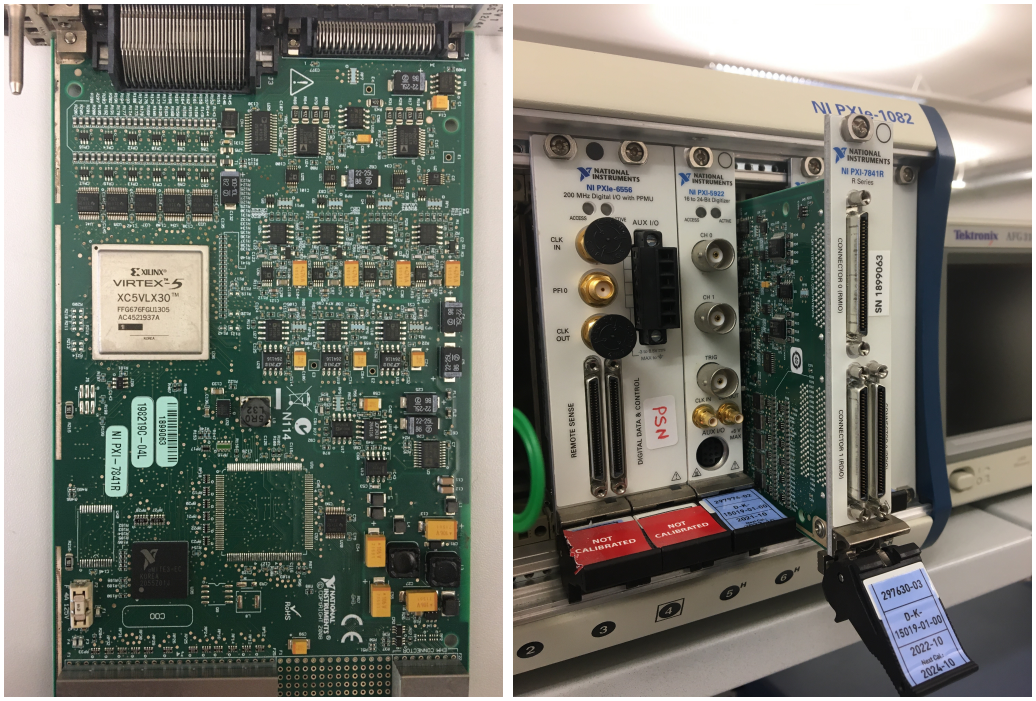


Figura 3.4: Modulo PXI 7841R

3.2.1 Il dispositivo FPGA

Il modulo evidenziato in Fig. 3.4 incorpora il dispositivo FPGA **Virtex-5**.
L'elemento centrale di qualsiasi FPGA è costituito da una matrice di blocchi logici configurabili detti **CLB** (Configurable Logic Blocks) [6], tali blocchi sono connessi tra loro attraverso interconnessioni programmabili dall'utente in modo tale da poter implementare semplici funzioni logiche o aritmetiche. Ai margini di tale matrice vi sono i blocchi di ingresso/uscita, detti **IOB** (Input Output Block). L'interconnessione di più blocchi logici permette di realizzare circuiti di complessità maggiore.
I blocchi CLB sono composti da due o quattro celle logiche (logic cell) che eseguono le operazioni booleane. Ogni cella logica è solitamente composta da una o più **LUT**

(Look-Up Table) programmabili. Le LUT sono utilizzate per implementare funzioni booleane generalizzate e sono solitamente accompagnate da un registro. L'insieme di questi elementi rappresenta uno dei parametri più importanti nella valutazione delle potenzialità di un FPGA.

Negli anni, tuttavia, Xilinx ha modificato le caratteristiche dei blocchi logici configurabili (CLB) presenti sui loro dispositivi.

I nuovi CLB presenti sulle FPGA Virtex-5 risultano più flessibili e complessi di quelli standard descritti brevemente sopra, con l'aggiunta di diverse funzionalità.

Il CLB delle Virtex-5 è costituito da due unità funzionali (**slice**) che, per svolgere funzioni più complesse, possono essere utilizzate insieme o anche indipendentemente per implementare due funzioni più semplici. Le slice rappresentano i blocchi logici dell'FPGA che contengono le LUT ed i flip-flop, più alcuni circuiti dedicati all'implementazione di funzioni aritmetiche. Ciascun CLB è connesso ad una matrice di commutazione (Switch Matrix), che permette l'instradamento dei segnali, e ai CLB adiacenti in modo da realizzare operazioni aritmetiche più efficienti.

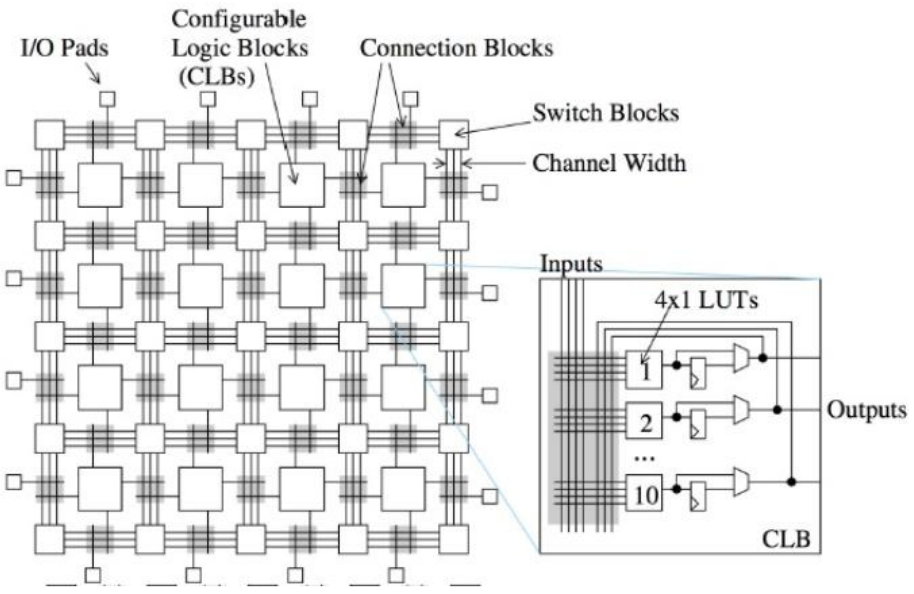


Figura 3.5: Struttura di una FPGA

I dispositivi FPGA presentano numerosi vantaggi. Essi sono programmabili direttamente dall'utente finale e ciò comporta la diminuzione dei tempi di progettazione, di verifica mediante simulazioni e di prova sul campo dell'applicazione. Dunque con tali componenti è possibile, semplicemente riprogrammando il dispositivo, correggere errori o apportare eventuali modifiche e per tale motivo sono molto utilizzati in ambito di prototipazione.
Tuttavia come aspetto negativo hanno l'elevato costo, soprattutto per applicazioni che ne richiedono un grande numero.

3.3 Altium Designer

Altium Designer [7] è un pacchetto software che integra tutti gli strumenti necessari per progettare e realizzare schematici e gerber files per la produzione di circuiti stampati. È sviluppato dalla società di software australiana Altium Limited.
Altium Designer offre un ambiente di progettazione unificato che consente agli utenti di avere una visione unica per ogni aspetto del processo di progettazione PCB, dallo schematico al layout PCB, fino alla documentazione di progettazione. Dunque è possibile accedere a tutti gli strumenti di progettazione in un unico ambiente intuitivo e completare l'intero processo di progettazione, fornendo in maniera rapida prodotti di alta qualità.

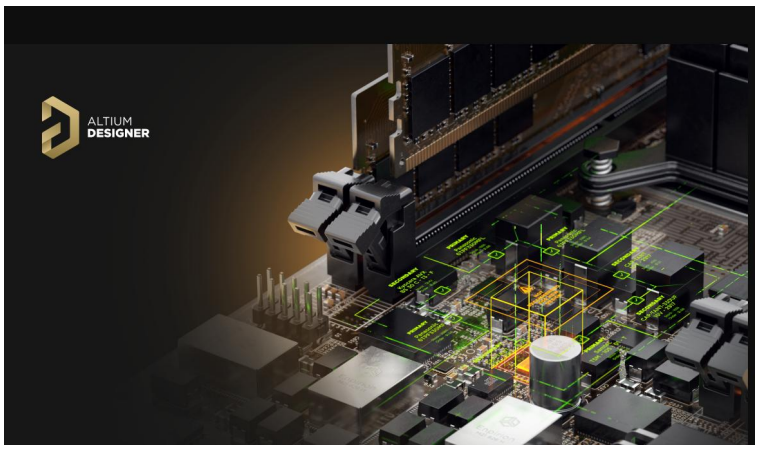


Figura 3.6: Altium Designer

La creazione di un progetto su Altium Designer prevede due step:

1. Progettazione dello **schematico**, in cui vengono aggiunti e connessi tra loro tutti i vari componenti che si intende utilizzare per la realizzazione del circuito;

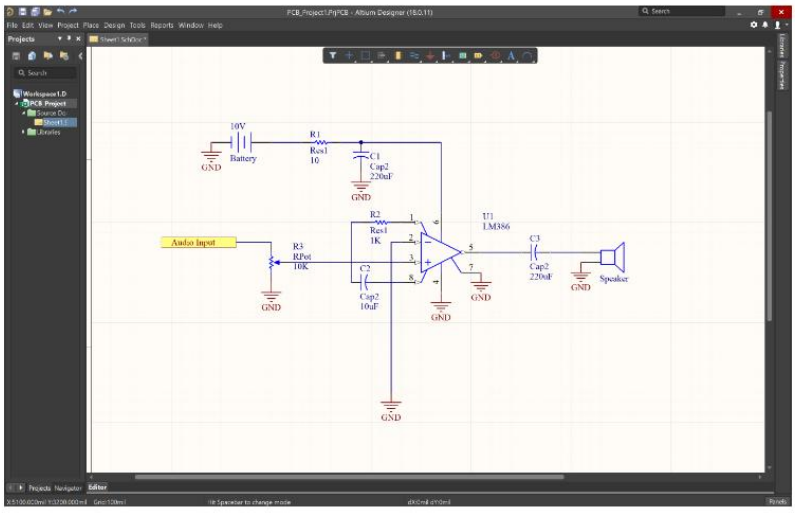


Figura 3.7: Primo step: creazione schematico

2. Importazione dei dati relativi al progetto schematico in uno schema **PCB** in modo da poter procedere con il layout.

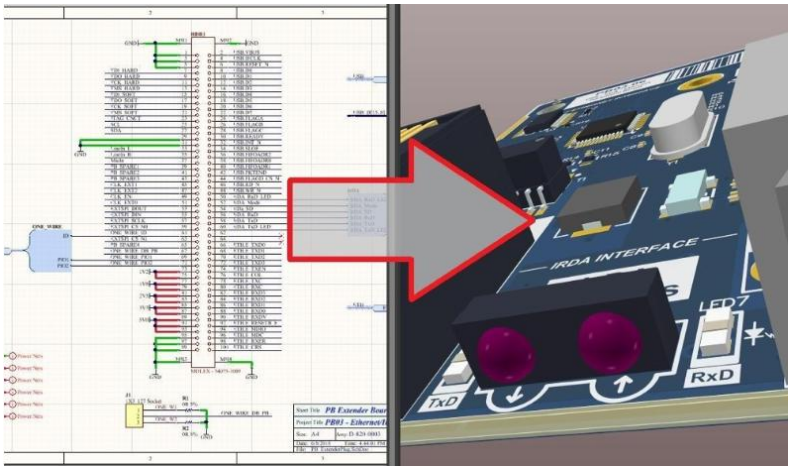


Figura 3.8: Secondo step: da progetto schematico a realizzazione PCB

Dopo il secondo step si procede con il collegamento delle tracce, il posizionamento corretto dei componenti sulla PCB, la creazione di un piano di massa per il collegamento al ground ed infine la produzione del progetto finale.

3.4 Bench di laboratorio

I laboratori di Infineon mettono a disposizione banchi di lavoro e varia strumentazione utilizzabile da ciascun utente. La seguente sezione descrive gli strumenti di misura utilizzati per la realizzazione di un banco di prova che servirà per testare il funzionamento della PCB collegata all'FPGA.

DC Power Supply

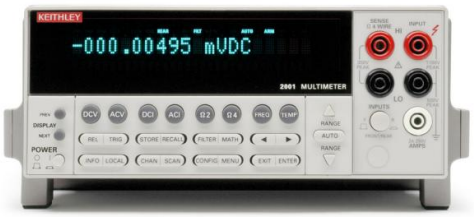
Il Toellner TOE8951 [8] è un alimentatore a singolo canale in grado di fornire fino a 400W e viene solitamente utilizzato al posto di una tipica batteria automobilistica a 12V. In tal caso permette l'accensione del dispositivo LDO da testare, fornendogli 12V in ingresso.

Sourcemeter

Il sourcemeter usato è il Keithley 2400, che è uno strumento elettronico in grado di generare correnti e tensioni e di misurarle contemporaneamente con elevata precisione.
Si differenzia rispetto al power supply per una migliore risoluzione e range di funzionamento: il sourcemeter è progettato per offrire una migliore capacità a bassa corrente rispetto al power supply e, per questo motivo, offre in genere intervalli operativi molto più ampi con una risoluzione maggiore [9]. In questo progetto, è stato utilizzato per fornire 5V all'enable dell'LDO e per fungere da carico elettronico attivo.

Multimeter

Il multimetro utilizzato è il Keithley 2001 che ha la funzione di misurare diverse grandezze.
Nello specifico, è stato collegato all'uscita dell'LDO per poter misurare i valori di tensione forniti in uscita dall'LDO.



Oscilloscopio

L'oscilloscopio è uno strumento di misura elettronico che consente di visualizzare, su un grafico bidimensionale, l'andamento nel dominio del tempo dei segnali elettrici ed effettuare misure a lettura diretta di tensione, rappresentata sull'asse verticale, e periodo, con il tempo rappresentato sull'asse orizzontale.
È stato utilizzato l'oscilloscopio Lecroy HDO6054: esso presenta 4 canali, una banda passante fino ad 1 GHz ed un'interfaccia utente semplice ed intuitiva.

Waveform Generator

Il generatore di funzioni/forme d'onda arbitrarie 33250A di Agilent Technologies adotta tecniche di sintesi digitale diretta per creare uscite stabili e accurate con qualsiasi forma d'onda, fino a una risoluzione di appena 1 μHz di frequenza [10].
In tal caso, tale dispositivo viene utilizzato per generare un impulso a partire dalla tensione di accensione tra il gate ed il source del mosfet (pari a 4V) che viene utilizzato per pilotare il carico passivo per effettuare i test relativi all'overcurrent.

Thermostream T-2500E

Il ThermoStream è un sistema portatile che fornisce aria secca e pulita per test di precisione della temperatura o condizionamento di componenti elettronici (circuiti integrati, MEMS, ricetrasmettitori o circuiti) e materiali. Nessun altro sistema è in grado di portare gli oggetti del test ad una determinata temperatura così velocemente e con un controllo preciso.

In particolare, il sistema di forzatura della temperatura del thermostream T-2500E, da -90° a 225°, offre funzionalità avanzate di test della temperatura. Il sistema passa dalla temperatura dell'aria da +125° a -55° in soli 4 secondi ed ha un'ampia gamma di temperature da -90° a +225° che soddisfa i requisiti degli ambienti di produzione e di progettazione. La refrigerazione meccanica a singolo stadio elimina la necessità di azoto liquido o di altri refrigeranti consumabili. Per garantire l'uso di aria pulita e secca, tale thermostream utilizza un essiccatore d'aria incorporato che rimuove il vapore acqueo e le particelle dalla fonte di aria compressa.

I sistemi di forzatura della temperatura sono collaudati sul campo in termini di durata, prestazioni ed affidabilità. Il sistema può, inoltre, essere utilizzato in ambienti di produzione tutto il giorno [11].

In tale progetto il thermostream viene utilizzato per effettuare i test per il controllo di overtemperature, portando l'LDO alle temperature di prova.

Capitolo 4

Preparazione setup FPGA

Il primo step per la realizzazione di questo progetto prevede il setup dell'FPGA e l'installazione di LabVIEW 2019: in particolare, è utile l'estensione LabVIEW FPGA Module in quanto consente di sviluppare ed eseguire il debug della logica hardware personalizzata in modo che sia possibile compilare e distribuire su hardware FPGA NI. Inoltre è opportuno installare, nel sistema collegato alla scheda e in tutti quelli in cui si vogliono sviluppare i vari VI, i drivers per R-series RIO devices, e i Xilinx compilation tools per ISE 14.7 nel sistema che eseguirà la compilazione (entrambi sono delle estensioni di LabVIEW).

4.1 Scrittura codice e problematiche riscontrate

Durante l'installazione di LabVIEW e LabVIEW FPGA, sono state riscontrate numerose problematiche. È stato necessario, pertanto, l'utilizzo del sintetizzatore Synopsys Synplify di FPGA in quanto l'ISE 14.7 non era in grado di compilare i codici scritti in SystemVerilog. Per poter utilizzare tale software bisogna avere accesso ad un Unix Personal Account. Nel caso in cui il codice fosse stato scritto in VHDL o Verilog, sarebbe stato possibile utilizzare anche l'ISE 14.7.
Durante la creazione dei CLIP (Component Level IP) e la compilazione del codice sono state individuate incompatibilità nell'utilizzo di Windows 10. A questo punto si sono presentate, quindi, due soluzioni adottabili:

1. Creare su VirtualBox una macchina virtuale con Windows 7, sulla quale installare LabVIEW 2019, LabVIEW FPGA Module, drivers per R-series RIO devices e i Xilinx compilation tools per ISE 14.7. In questo modo, è possibile generare il file XML delle CLIP seguendo lo wizard di LabVIEW. Questa alternativa, tuttavia, è sconsigliata a causa della difficoltà nel procurarsi una licenza per una versione obsoleta di Windows.

   In ogni caso, anche riuscendo ad ottenere una licenza per Windows 7, il problema che si presenterebbe sarebbe legato alla mancanza di aggiornamenti di sicurezza futuri per il sistema operativo, cosa che innalzerebbe inevitabilmente il livello di vulnerabilità del sistema. Inoltre, questo percorso consente solo la compilazione sul cloud della National Instruments, il che potrebbe portare alla violazione della confidenzialità dei progetti.

2. Creare su VirtualBox una macchina virtuale con CentOS (preferibilmente 9 o inferiore) sulla quale installare solo i Xilinx compilation tools per ISE 14.7 e le sue dipendenze di software e, successivamente, installare sulla macchina host il LabVIEW Compile Farm Toolkit. In questo caso, il file XML della CLIP dovrà essere scritto manualmente, seguendo le istruzioni della guida di compilazione del file XML per inserimento delle CLIP.

   Poi, per la compilazione del codice per l'FPGA, si sceglie l'opzione "connect to a network compile server". Questa, infatti, è l'alternativa consigliata in quanto la compilazione viene eseguita in un server locale, pertanto non ci sono problemi di licenza o sicurezza associati al sistema operativo della macchina virtuale.

4.2 Compilazione codice

Dopo la scrittura del codice, il programma deve essere compilato. Il compilatore FPGA è una raccolta di algoritmi che, attraverso una serie di elaborazioni, genera un layout ottimizzato sul chip, configurabile in molteplici modi e con milioni di elementi. Le compilazioni FPGA si compongono di tre fasi:

- La prima fase è l'**analisi e sintesi**, dove il codice scritto in LabVIEW è analizzato e successivamente viene generato il codice equivalente VHDL. Il sintetizzatore poi analizza i costrutti del linguaggio VHDL e riconosce i componenti ad alto livello utilizzati (contatori, multiplexer, decoder, etc.).

- La seconda fase è il **place & routh**, dove il compilatore identifica tutti gli elementi di memoria di alto livello (registri, contatori, shift-register) e li riconduce ad elementari Flip-Flop. Identifica le funzioni di trasferimento (equazioni booleane), ovvero i percorsi combinatori tra registri, tra input e registri, tra registri e output, tra input e output e le ottimizza eliminando quelle superflue. Infine, mappa le equazioni sulle Look-Up Table delle celle logiche.

- La terza fase prevede la generazione del programma e del **bitfile** (eseguibile).

La compilazione [12] in LabVIEW FPGA avviene grazie a tre componenti fondamentali: il computer di sviluppo, il *Compile Server* e un gran numero di *Compile Worker*. Facendo clic sul tasto *Esegui* del VI FPGA sono generati dei file intermedi che sono inviati, attraverso una comunicazione Web, al Compile Server.
Il *Compile Server* accetta i lavori di compilazione da uno o più sistemi di sviluppo FPGA LabVIEW e cerca Compile Worker disponibili per la creazione di file compilati (bitfile). Se non sono disponibili Compile Worker, il server mette il file da compilare in una coda fino a quando un Worker diventa disponibile.
Il *Compile Worker* si occupa della generazione del bitfile per la configurazione dell'FPGA, sfruttando la presenza degli strumenti di compilazione Xilinx installati per la sintesi di progettazione FPGA (VHDL), la mappatura e il placing e routing.

Generato il bitfile, il Compile Worker invia il file al Compile Server che a sua volta lo rimanda al computer di sviluppo.

Il vantaggio di questa architettura del sistema di compilazione è che supporta più *Compile Worker* e pertanto garantisce una compilazione di un gran numero di file con tempi di attesa minimi.

Figura 4.1: Architettura del sistema di compilazione di LabVIEW FPGA

4.3 Realizzazione progetto LabVIEW FPGA

I passi per la realizzazione di un progetto in LabVIEW FPGA sono:

1. **Creazione IP**: inizialmente viene prodotto il codice HDL in SystemVerilog per poter essere programmato sull'FPGA tuttavia LabVIEW non permette l'importazione di tale codice direttamente sulla piattaforma menzionata, pertanto viene richiesto un wrapper in VHDL che contiene gli ingressi e le uscite collegati soltanto alla dichiarazione dei componenti. Inoltre, il codice in SystemVerilog deve essere sintetizzato in una netlist EDIF su Synplify prima di poter essere utilizzato. Dopo aver creato il progetto ed il file XML sulla macchina virtuale, si procede con l'importazione del CLIP che dovrà essere inserito nell'FPGA.

   Nella Project Explorer sarà presente l'IP le cui porte potranno essere trascinate sullo schema a blocchi di LabVIEW.

2. **Creazione DMA-FIFO**: il blocco FIFO (First In-First Out) è stato utilizzato per eseguire la comunicazione tra l'FPGA e l'host. Tali blocchi sono utilizzati per effettuare la comunicazione asincrona tra domini di clock diversi.

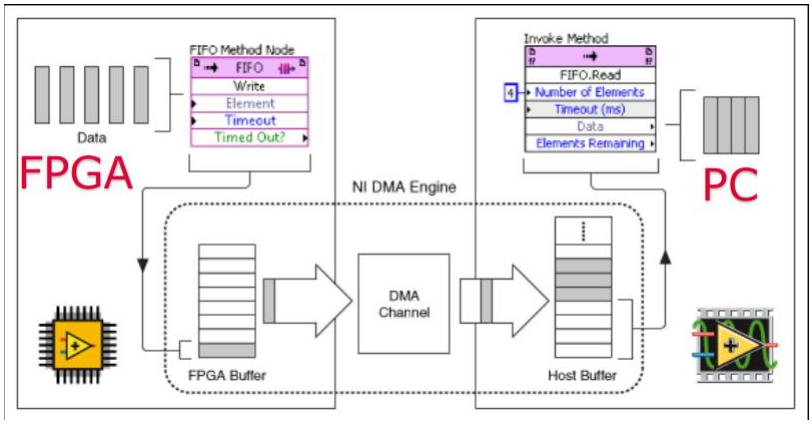


Figura 4.2: DMA-FIFO nel metodo 'Write'

Per la trasmissione dei dati dall'FPGA all'host viene inserito il blocco FIFO (nell'FPGA main) nel metodo '**Write**' che contiene i seguenti parametri:

- Element: elementi di trasmettere;

- Timeout: tempo massimo di attesa per ricevere un elemento, nel caso in cui fosse inserito in un timed loop deve essere pari a 0;
- Timed Out?: valore booleano che indica se il metodo è andato o meno in timed out. Può essere lasciato scollegato qualora non si desiderasse avere informazioni sul timeout.

Per la ricezione dei dati sull'host viene inserito, nel PC main, il blocco 'Invoke Method'. Una volta collegato alla reference della VI, potrà essere impostato nel metodo '**Read**' il quale fa riferimento alla FIFO creata nell'FPGA. Tale FIFO contiene i seguenti parametri:

- Number of Elements: numero di elementi da leggere;
- Timeout (ms): tempo massimo di attesa per ricevere un elemento;
- Data: uscita dei dati ricevuti in un array del tipo di dati scelto;
- Elements Remaining: elementi rimanenti nel buffer dell'host dopo la lettura.

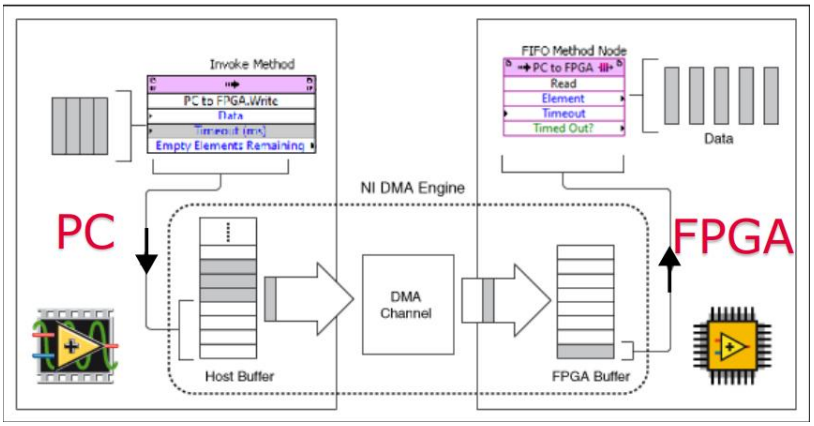


Figura 4.3: DMA-FIFO nel metodo 'Read'

Per la comunicazione tra l'host e l'FPGA, invece, vengono invertiti i metodi delle FIFO. Il primo blocco inserito nell'FPGA main fa riferimento al metodo 'Read' e, l'unica differenza rispetto al metodo Write che fa riferimento all'FPGA, è che il blocco 'Element' è trattato come un'uscita e non come un ingresso.

Per la trasmissione dei dati dall'host viene inserito, nel PC main, il blocco 'Invoke Method'. Una volta collegato alla reference della VI, potrà essere impostato nel metodo '**Write**' che fa riferimento alla FIFO creata nell'FPGA. Tale FIFO contiene i seguenti parametri:

- Data: ingresso dell'array dei dati da trasmettere;
- Timeout: tempo massimo di attesa per ricevere un elemento;
- Empty Elements Remaining: indica i posti vuoti nel buffer che possono essere ancora riempiti.

Per l'inserimento dei blocchi FIFO nell'FPGA e la creazione delle rispettive FIFO, si procede come mostrato in figura:

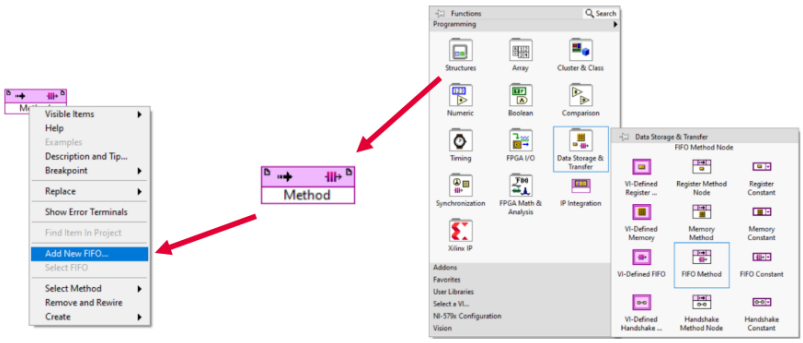


Figura 4.4: Creazione ed inserimento blocchi FIFO nell'FPGA

4.4 Test eseguiti

Dopo aver terminato la parte di setup e di creazione dei vari blocchi, sono stati eseguiti diversi test per capire il funzionamento tra host e FPGA.

4.4.1 Test 1: Test preliminare di funzionamento

a) **Creazione VI-FPGA**

Per permettere il collegamento tra i blocchi LabVIEW e la IP è stata creata la VI 'FPGA main' la quale viene compilata per poter essere programmata sull'FPGA. Una volta creato ed importato il Component-Level IP si aprirà un'icona che mostrerà i suoi terminali disponibili. Tali terminali verranno poi trascinati all'interno del Block Diagram per fare i dovuti collegamenti.

La VI mostrata nella figura sottostante è stata la prima prova per permettere la comunicazione tra l'FPGA ed il PC.

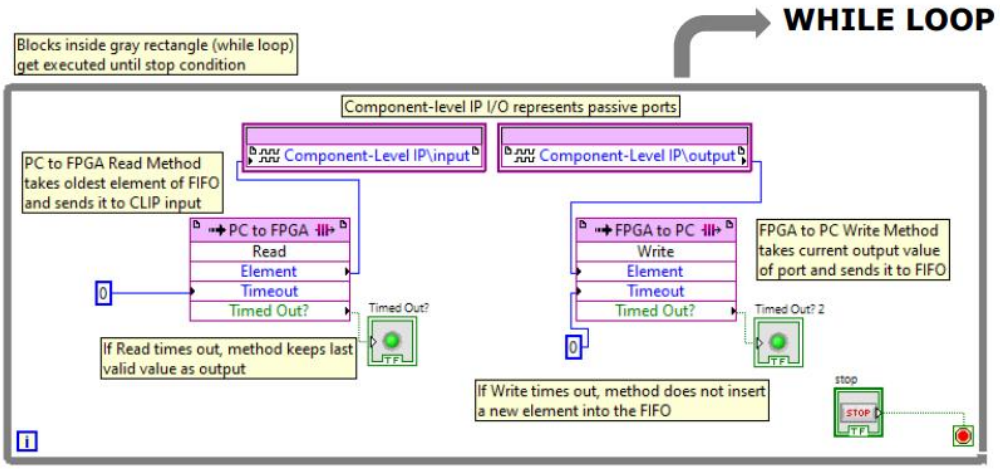


Figura 4.5: FPGA main (test 1)

In tal caso, il CLIP utilizzato è un blocco combinatorio che esegue la funzione NOT su un numero in ingresso per poterlo poi ricevere, in uscita, sul PC.

I blocchi dei CLIP sono l'input e l'output della IP che viene trattata come una scatola nera. I blocchi FIFO sono utilizzati per la lettura (Read) e scrittura

(Write) in forma asincrona (dati trasmessi in modo intermittente anziché in un flusso costante) tra il PC e l'FPGA. Nell'FPGA, tali blocchi FIFO sono considerati dei buffer in block RAM che vengono riempiti e svuotati in base alla lettura e/o scrittura.

Ad ogni iterazione del while loop, la FIFO Read prende uno dei valori del buffer e lo trasmette al CLIP di input, simultaneamente la FIFO Write registra il valore rilevato dal CLIP di output e lo inserisce all'interno del buffer. Infine, il buffer di Write trasmette i valori acquisiti al pc mentre il buffer di Read riceve nuovi valori.

b) **Creazione VI-PC**

Per l'inserimento di altre VI di elaborazione dei dati da inviare all'FPGA o da ricevere dall'FPGA, si usa la VI mostrata nella figura che segue.

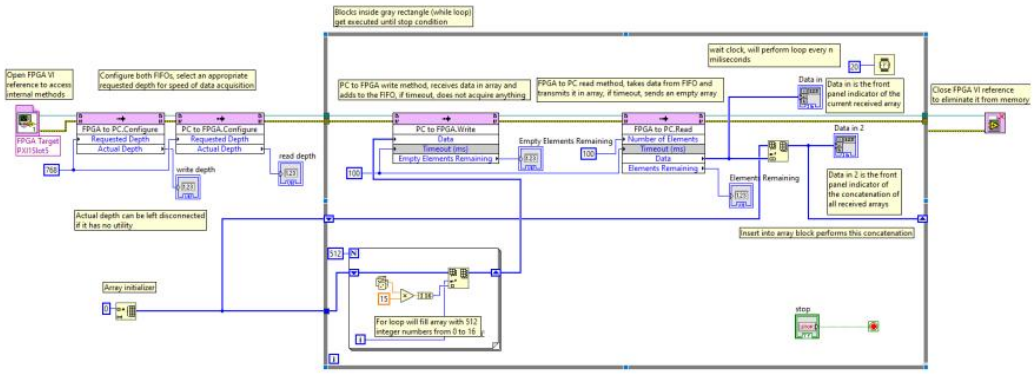


Figura 4.6: PC main (test 1)

Il programma mostrato in Fig. 4.6 ha lo scopo di generare un array di numeri casuali, inviarli all'FPGA per poi poterli ricevere sul PC.

Fuori dal loop, dopo aver inserito il blocco 'Open FPGA VI Reference' che consente di accedere alle funzionalità di comunicazione dati dell'FPGA, si esegue la configurazione dei blocchi FIFO in cui viene selezionato il parametro relativo alla lunghezza del buffer del PC.

Dentro al loop, viene generato un array di numeri casuali da 0 a 15, per scopi di test, che viene inviato alla FIFO Write ad ogni iterazione del loop. Successiva-

mente la FIFO Read riceve l'array dei numeri elaborati dall'FPGA e lo inserisce in due array che vengono visualizzati sul front panel di LabVIEW.

Non appena viene premuto il pulsante di stop, il loop termina ed il programma viene chiuso.

4.4.2 Test 2: Macchina a stati

a) **Creazione VI-FPGA (test 2)**

Tale test è stato eseguito per capire il funzionamento della FIFO con l'inserimento di una macchina a stati, la quale prende un numero di 4 bit in ingresso su un CLIP denominato 'data in', li riordina e trasmette il risultato in uscita su un CLIP denominato 'data out'.

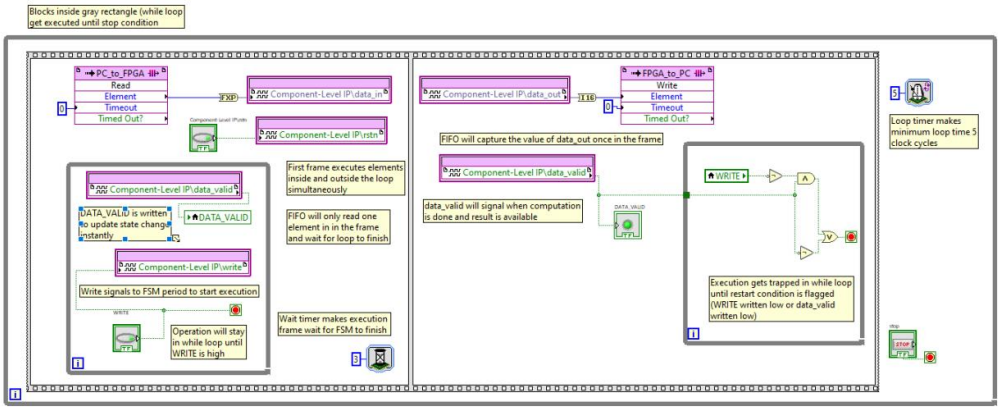


Figura 4.7: FPGA main (test 2)

In tal caso, la FPGA main è composta da un while loop al cui interno troviamo una Flat Sequence Structure. Tale struttura è divisa in due parti: la prima contiene la parte di ricezione e la seconda la parte di trasmissione. La parte di ricezione è composta da due blocchi che vengono eseguiti in parallelo: il primo verifica, tramite un while loop, se il pulsante di Write viene premuto e simultaneamente agisce sull'indicatore 'data valid' per resettarlo; il secondo blocco riceve un dato in ingresso e lo trasmette al pin 'data in' del CLIP. Inoltre,

in questa parte è impostato un timer di attesa pari a 3 cicli di clock per lasciare che i risultati della macchina a stati siano disponibili sulla parte di trasmissione nella stessa iterazione del while loop.

La parte di trasmissione è composta, invece, da un pin 'data out' che ha il compito di inviare i dati alla FIFO e dal pin 'data valid' che indica che l'operazione è andata a buon fine. La macchina a stati è impostata in modo tale che l'operazione inizi con la ricezione del segnale di Write che rimane alto per tutta la durata dell'elaborazione. Una volta terminata l'elaborazione, la macchina a stati emette il segnale 'data valid' finchè il segnale di Write diventi basso. Il secondo blocco della parte di trasmissione è costituito da un while loop che blocca il proseguimento del programma finchè il segnale di write non diventa basso.

In ogni caso, se 'data valid' non diventa alto il programma esegue il while loop una sola volta ed esce per aspettare l'arrivo di nuovi dati.

b) **Creazione VI-PC**

Il PC main in figura sottostante ha lo scopo di generare un array di numeri casuali, inviarli all'FPGA per poi poterli ricevere sul PC.

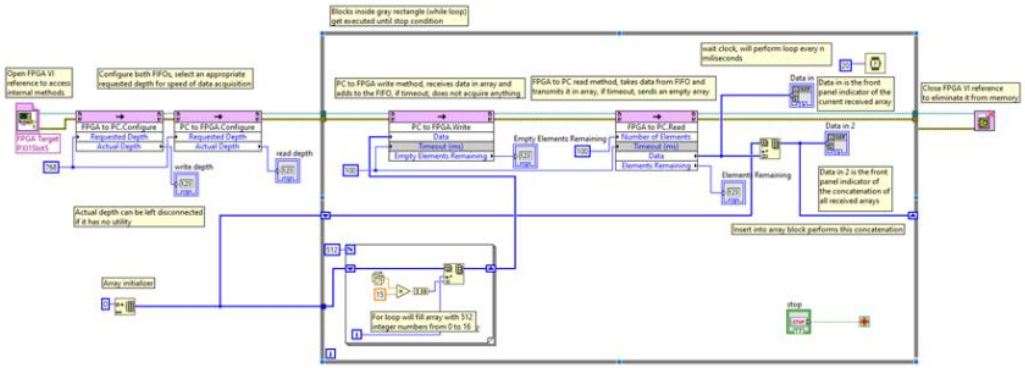


Figura 4.8: PC main (test 2)

4.4.3 Test 3: Trasmissione dati dall'FPGA al PC

In questo test, è stata creata una CLIP con un contatore incrementale che permette di valutare la velocità di trasmissione dei dati dall'FPGA al PC e controllare l'esistenza o meno degli errori di trasmissione.

a) **Creazione VI-FPGA**

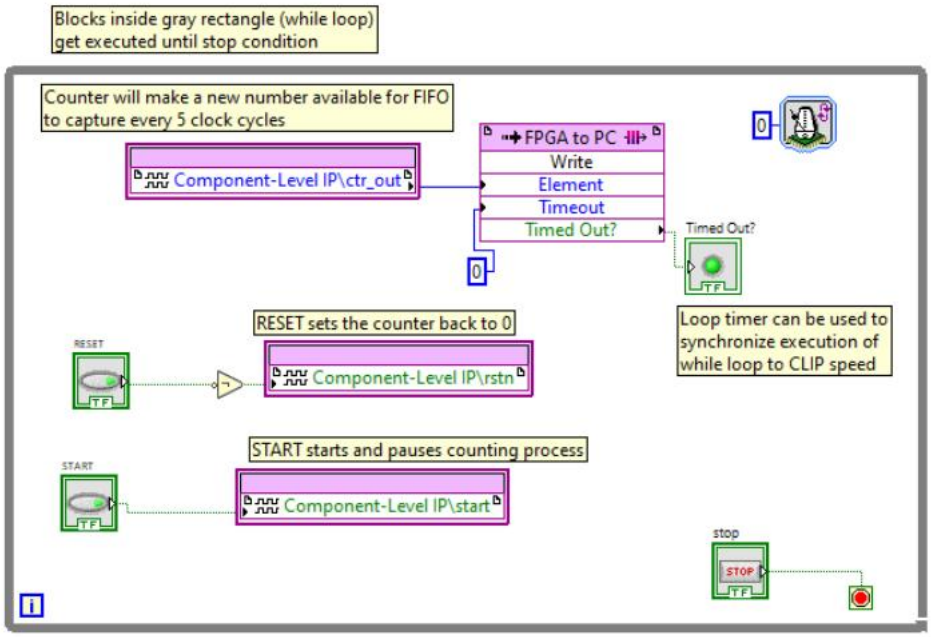


Figura 4.9: FPGA main (test 3)

Tale VI ha lo scopo di prendere i dati della CLIP e trasmetterli alla FIFO. È costituita da un while loop al cui interno troviamo i due pin in ingresso per il reset e lo start: il reset porta il contatore a 0 quando viene premuto mentre il tasto start è in grado di avviare o mettere in pausa l'operazione. Il segnale "counter out" rappresenta l'uscita dei numeri del contatore: ogni volta che viene eseguito il while loop, la FIFO legge il valore attuale in uscita dal "counter out" e lo inoltra al PC.

Il blocco di loop timer, invece, serve per rallentare la frequenza di esecuzione del ciclo, in modo da sincronizzare l'esecuzione del blocco della FIFO all'uscita di dati dal CLIP o semplicemente impostare un periodo di esecuzione fisso:

ciò può essere impostato sia con cicli di clock, per sincronizzazioni precise, sia con millisecondi o microsecondi, per sincronizzazioni indipendenti dal clock utilizzato.

b) **Creazione VI-PC (PC_main_repeat)**

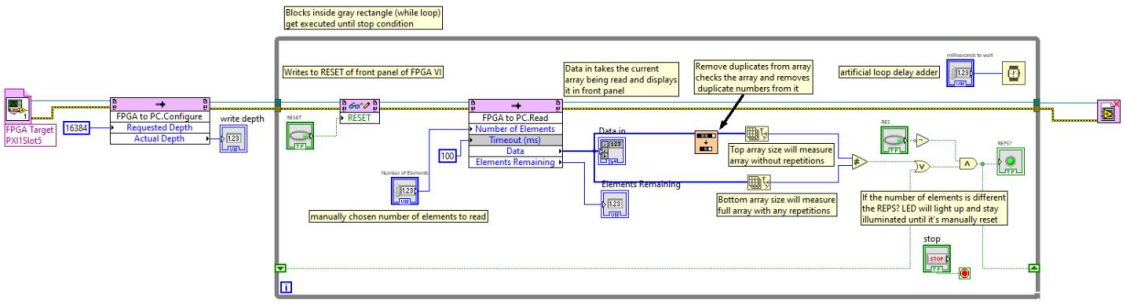


Figura 4.10: PC_main_repeat

Tale PC_main_repeat è stata creata per leggere gli elementi della FIFO e verificare che non ci siano numeri ripetuti al suo interno in quanto, nei test preliminari, sono stati riscontrati dei problemi di sincronizzazione tra il contatore del'FPGA e le FIFO che portavano ad avere dei valori che venivano trasmessi due volte (CLIP più lento della FIFO). Una volta che l'array di dati viene ricevuto dal buffer della FIFO, i dati sono duplicati ed inviati da un lato nel blocco 'Remove duplicates from 1D array' che ha il compito di rimuovere i duplicati e dal quale, tramite il blocco 'Array size', viene misurata la dimensione dell'array, e dall'altro lato viene misurata semplicemente la dimensione dell'array di dati proveniente direttamente dalla FIFO.

Dai risultati ottenuti da ambo i lati viene fatto un confronto di disuguaglianza che permette di visualizzare, tramite un LED, se il programma presenta o meno delle ripetizioni di dati.

c) **Creazione VI-PC (PC_main_skip)**

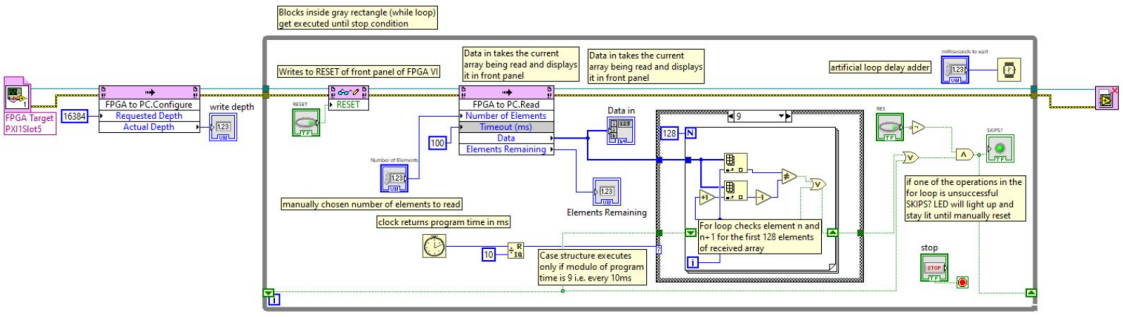


Figura 4.11: PC_main_skip

Tale VI è stata realizzata per leggere gli elementi della FIFO e verificare che non ci fossero dei salti di elementi al suo interno in quanto, nei test preliminari, si è notato che sia quando il buffer del PC era pieno e sia quando il CLIP era più veloce della FIFO, questa non riusciva ad acquisire tutti i valori generati.

Una volta che l'array di dati viene ricevuto dal buffer della FIFO, i dati vengono inviati ad una case structure utilizzata per fare un campionamento dei dati: ogni 10 ms viene preso un campione dei dati in uscita e si esegue, nei primi 128 elementi, il confronto tra l'n-esimo valore e l'(n+1)esimo elemento e quest'ultimo viene, a sua volta, sottratto di uno e confrontato con l'n-esimo valore. Se i due valori in uscita sono diversi, si accende un LED alla fine del ciclo che evidenzia la presenza di un dato saltato.

Risultati e considerazioni implementative

Partendo da una frequenza di clock dell'FPGA di 40 MHz, sono stati ottenuti i seguenti risultati e considerazioni:

- La velocità massima di trasferimento dei dati è pari a 64 MB/s;
- La FIFO è in grado di gestire un unico dato ogni 5 cicli di clock. Se i dati fossero generati impostando una frequenza più alta, si rischierebbe di avere perdita di dati;

- È opportuno impostare il buffer del PC ad un valore più elevato del buffer dell'FPGA in modo che, nel caso in cui ci fossero ritardi nell'elaborazione dei dati, non si riscontrino problemi di dati non ricevuti. In tale test, infatti, si è impostato un valore del buffer del PC quattro volte più elevato rispetto a quello dell'FPGA ottenendo un'acquisizione corretta di tutti i dati anche durante l'esecuzione del ciclo for.

4.4.4 Test 4: Trasmissione dati dal PC all'FPGA

In questo test, è stata creata una CLIP che permette di valutare la velocità di trasmissione dei dati dal PC all'FPGA.

a) **Creazione VI-FPGA**

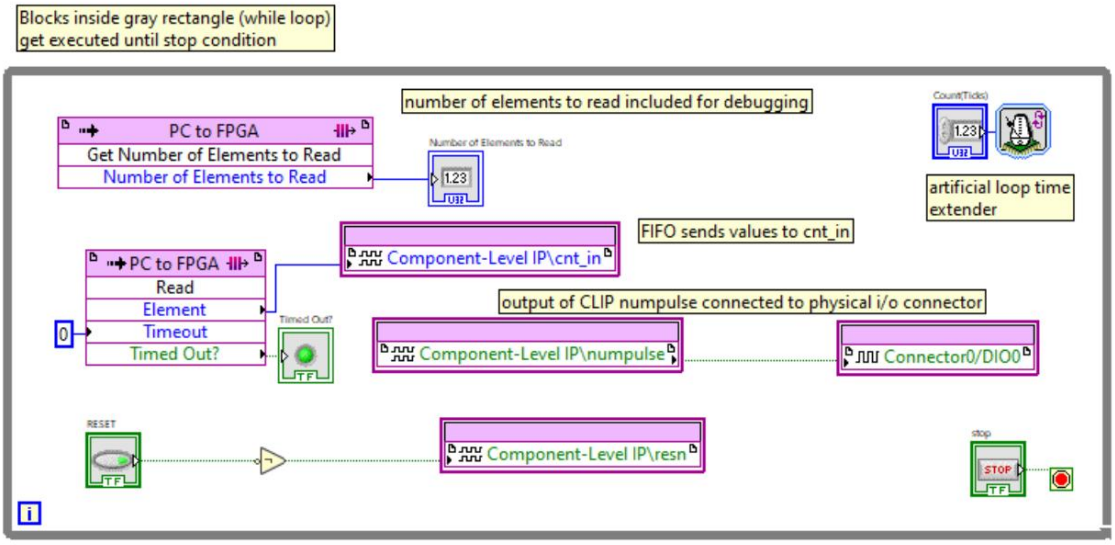


Figura 4.12: FPGA main (test 4)

Tale VI ha lo scopo di prendere i dati della FIFO e trasmetterli al CLIP. È costituita da un while loop al cui interno troviamo il pin in ingresso per il reset che ripristina il sistema alle condizioni di partenza quando viene premuto ed il segnale "counter in" che, invece, rappresenta l'ingresso dei numeri da elaborare: ogni volta che viene eseguito il while loop, la FIFO legge il valore più vecchio

del buffer dell'FPGA e lo inoltra al CLIP. Ogni volta che il CLIP riceve un numero che non sia zero, invia un segnale alto tramite l'uscita "numpulse" che viene connessa ad un pin fisico DIO0 grazie al quale è possibile visualizzare i relativi segnali con un oscilloscopio.

b) **Creazione VI-PC**

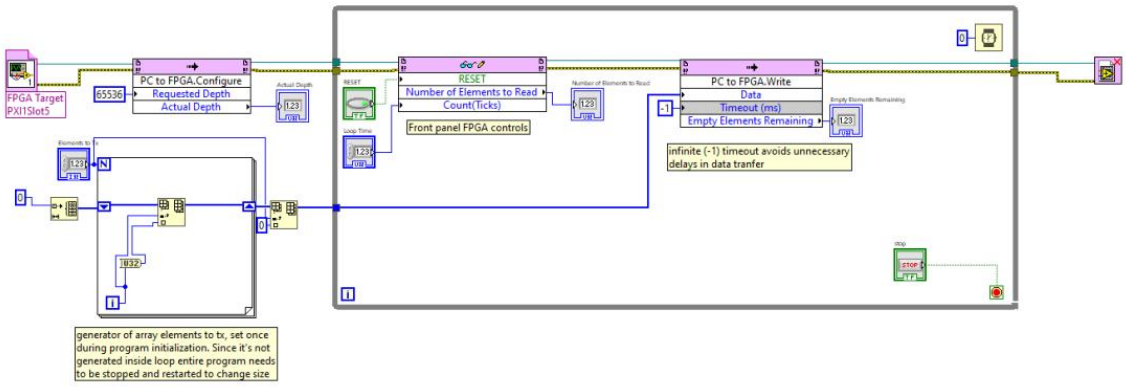


Figura 4.13: PC main (test 4)

Tale VI è stata creata per generare un array di numeri da inviare al buffer della FIFO nel PC presente all'interno del while loop. La generazione dell'array è ottenuta tramite un ciclo for posto fuori dal while loop affinché non ci siamo problemi di timing con la trasmissione del segnale: l'array è costituito da numeri sequenziali da 0 ad N-1 (N è impostato sul front panel) e con uno 0 alla fine.

Risultati e considerazioni implementative

Partendo da una frequenza di clock dell'FPGA di 40 MHz, sono stati ottenuti i seguenti risultati e considerazioni:

- La velocità massima di trasferimento dei dati è pari a 35MB/s;
- La FIFO è in grado di gestire un unico dato ogni 9 cicli di clock;
- I risultati sperimentali hanno mostrato che, una volta svuotato il buffer dell'FPGA, l'ultimo elemento trasmesso rimane in uscita fino alla successiva ricezione

di dati. Pertanto, è opportuno inserire degli elementi di header e footer per evidenziare l'inizio e la fine del pacchetto di dati;

- Per ottimizzare la trasmissione dei dati in modo da non avere periodi di inattività sulla FIFO, conviene impostare il timeout del metodo pari a -1 (valore che indica assenza di timeout) per permettere l'invio dei dati non appena vengono ricevuti.

Capitolo 5

Design PCB

Una volta appurata la bontà del lavoro svolto sull'FPGA (descritto nel capitolo precedente) si è passati a pensare a come replicare il comportamento di un generico regolatore LDO integrato in un PMIC.
La scelta è ricaduta subito sul design di una PCB che avesse come elemento centrale un LDO e che presentasse alcuni elementi circuitali di contorno per l'implementazione di alcuni controlli necessari per chiudere il loop con la logica integrata nell'FPGA.

5.1 Scelta dell'LDO

Il primo passo prevede la selezione di un LDO che presenti caratteristiche in linea con quelle del generico PMIC LDO di cui si intende replicare il comportamento: si è quindi scelto l'LDO **TLE42764DV** per la presenza del pin VA (Voltage Adjust Input) di adjustable che permette di regolare la tensione d'uscita.

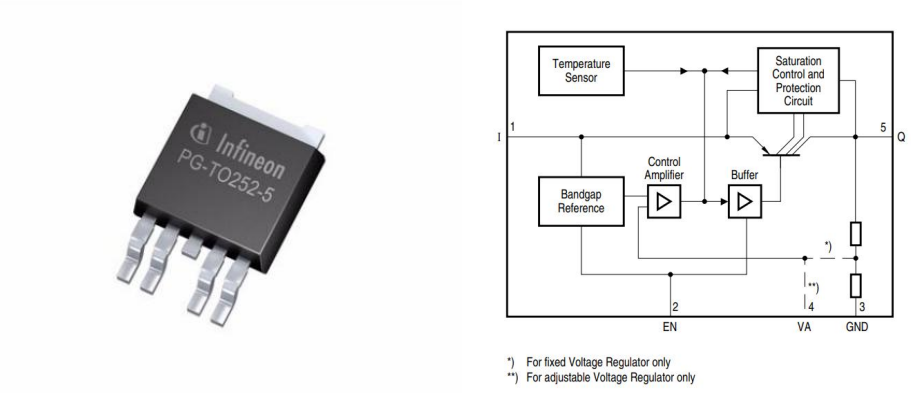


Figura 5.1: LDO TLE42764DV

Il TLE42764DV è un regolatore di tensione integrato a basso drop out usato per correnti di carico fino a 400 mA. La tensione d'ingresso arriva fino a 40 V e viene regolata ad una tensione regolabile o fissa di 5 V con una precisione di ±2%.
Il dispositivo è progettato per applicazioni automobilistiche pertanto è protetto da sovraccarichi, cortocircuiti e condizioni di overtemperature grazie all'implementazione della limitazione della corrente di uscita e dal circuito di spegnimento per overtemperature[13].
Il TLE42764DV può essere utilizzato in tutti i casi che richiedono una tensione stabile tra 2.5V e 20V.
Tale dispositivo presenta 5 pin, come si può osservare in figura 5.1:

- **Input (I)**: pin di ingresso;
- **Enable Input (EN)**: pin che abilita l'IC (integrated circuit) quando si ha un segnale di ingresso ad alto livello o che disabilita l'IC quando si ha un segnale di ingresso a basso livello;
- **Ground (GND)**: pin di ground;
- **Voltage Adjust Input (VA)**: pin che permette di avere una tensione regolabile;
- **Output (Q)**: pin di uscita.

Lo step successivo prevede l'implementazione di una circuiteria esterna all'LDO per il controllo dei segnali di 'undervoltage' (**UV**), 'overvoltage' (**OV**), 'temperature warning' (**TW**), 'temperature shut-down' (**TS**) e 'overcurrent' (**OVC**).
La board realizzata, pertanto, presenta **6** uscite (LDO output, OV, UV, TW, TS, OVC) e **2** ingressi (LDO input, enable).

5.2 Schematico per fornire alimentazione

Ogni circuiteria esterna all'LDO ha bisogno di **5 V** in ingresso per funzionare, pertanto c'è la necessità di realizzare uno schematico che regoli i 9V, che sono forniti esternamente alla board, in 5 V da dare in ingresso ai vari circuiti esterni all'LDO.
Nella figura sottostante viene mostrato lo schematico utilizzato per realizzare quanto precedentemente descritto.

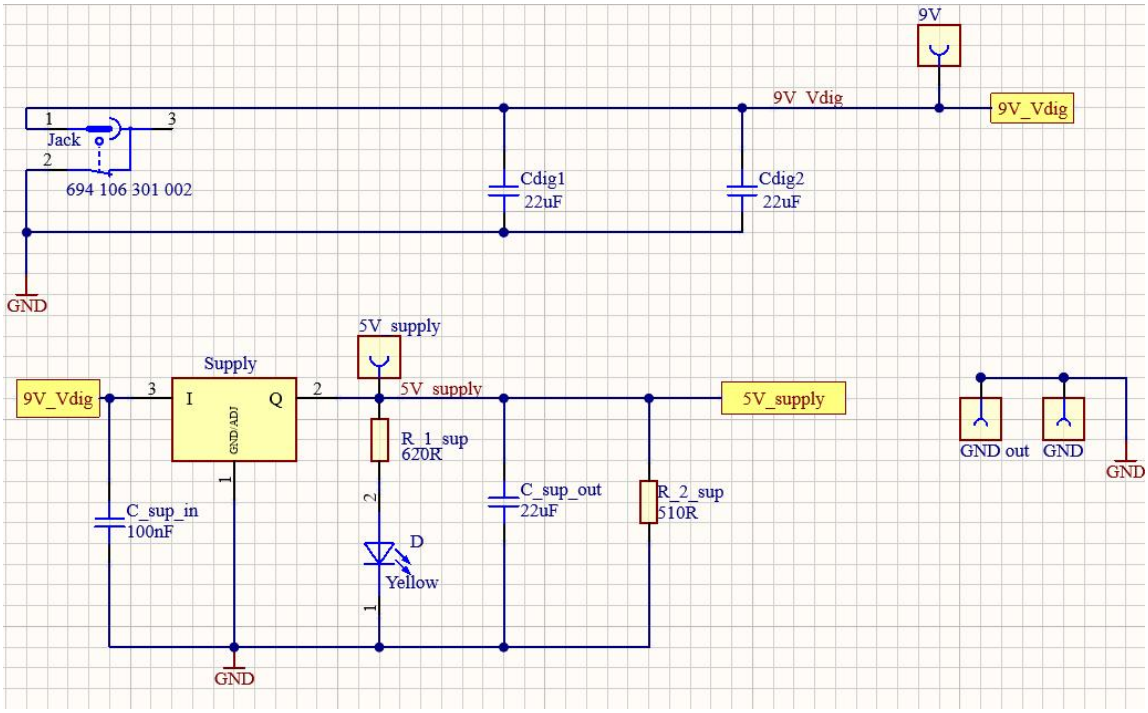


Figura 5.2: Schematico per supply

I 9V vengono forniti in ingresso al regolatore di tensione TLE4284.
Tale regolatore presenta due modalità di utilizzo: una per ottenere una tensione d'uscita fissa e l'altra per avere una tensione di uscita regolabile attraverso il pin di adjustable; in tal caso in uscita si vuole un valore fisso a 5 V pertanto è utilizzata la prima modalità operativa.

5.3 Circuiteria esterna all'LDO

In questa sezione vengono analizzati i vari circuiti elettronici per il controllo dei segnali citati precedentemente.

5.3.1 Segnale di enable

Da datasheet si ha che l'FPGA può emettere solo 3.3 V dai suoi pin di I/O digitali, mentre la tensione minima che l'interruttore di enable dell'LDO riconosce come segnale alto è di un valore di 3.5 V. Questo disallineamento provoca una situazione in cui l'FPGA non è in grado di abilitare l'LDO in maniera indipendente: per questo motivo, è stato aggiunto un comparatore di tensione sul connettore di ingresso del segnale di enable in modo da diventare alto a 2.4V.
Nella figura sottostante viene mostrato lo schematico realizzato su Altium Designer per il segnale di enable e per regolare la tensione in uscita dall'LDO.

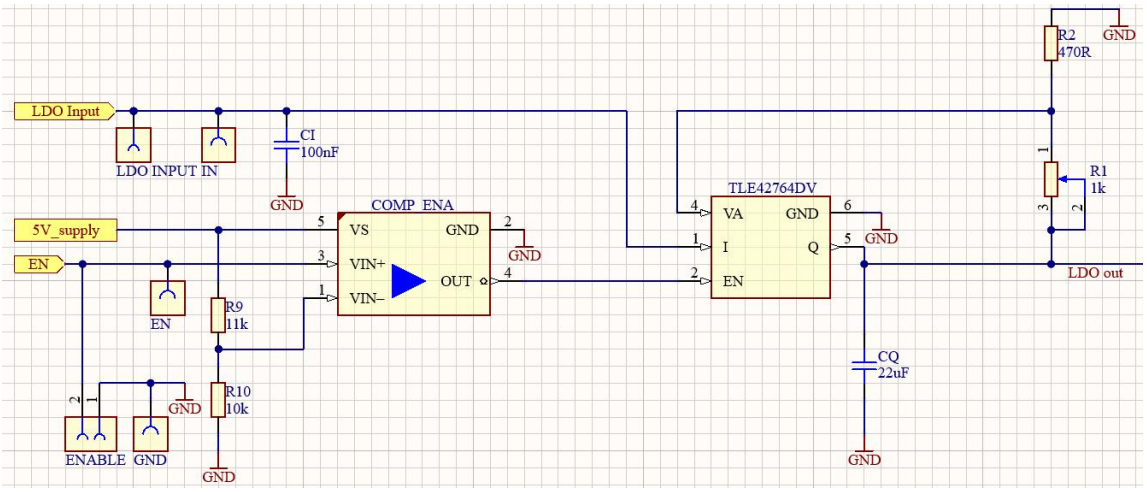


Figura 5.3: Circuito esterno all'LDO per segnale di enable e per regolare la tensione in uscita dall'LDO

È possibile notare come in ingresso sia stato posizionato un comparatore di tensione, esso è un circuito in grado di trasformare un segnale analogico, applicato al suo ingresso, in due soli possibili livelli di uscita: un livello alto e uno basso. Il suo principio di funzionamento si basa sul confronto tra il segnale analogico e una tensione di riferimento [14].

In tal caso viene confrontato il segnale fornito all'enable (V_{in}^{+}) con una tensione di riferimento pari a circa **2.4V** (V_{in}^{-}), valore ottenuto dal dimensionamento delle due resistenze R_9 ed R_{10} che formano un partitore di tensione, per abilitare l'LDO quando si verifica la condizione $V_{in}^{+} > V_{in}^{-}$.

5.3.2 Tensione in uscita dall'LDO regolabile

Dopo aver selezionato l'opportuno LDO ed aver realizzato il circuito esterno all'LDO per il segnale di enable, si procede con il dimensionamento delle resistenze per realizzare un partitore di tensione che renda l'uscita dell'LDO regolabile. Come mostrato in Fig. 5.3, il segnale in uscita dall'LDO è connesso al partitore di tensione, il quale è formato da una resistenza fissa di valore pari a 500Ω ed un potenziometro di 1kΩ per fornire **5 V** in uscita.

Con $R_1 = 1$ kΩ ed $R_2 = 500$ Ω (in Fig. 5.3 si ha R_2 di un valore pari a 470 Ω poiché su Altium non era presente il valore corretto) si ottengono, infatti, esattamente 5 V in uscita mentre per avere un'uscita variabile basterà regolare il potenziometro ad altri valori di resistenza.

5.3.3 Controllo per undervoltage (UV)

Nella maggior parte dei PMIC viene solitamente implementato un controllo di undervoltage per ogni loro post-regolatore. Questo segnale di undervoltage è molto utile in varie situazioni, ma soprattutto in fase di accensione in quanto la sua disattivazione comunica alla logica interna del PMIC la corretta accensione del post-regolatore.
Di seguito viene illustrata l'implementazione effettuata per questo progetto di tesi.

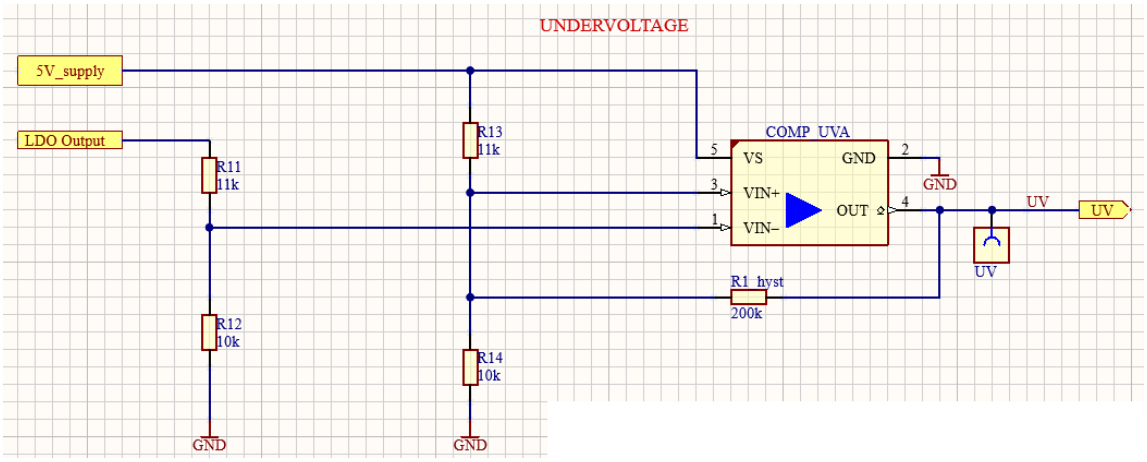
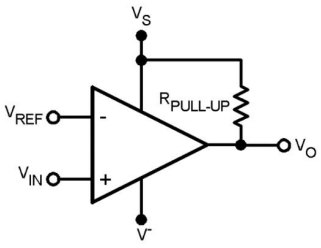


Figura 5.4: Circuito esterno all'LDO per undervoltage



È stato scelto il comparatore di tensione **LM397MF/NOPB** che presenta una configurazione 'open-collector' infatti, per tale motivo, è stato necessario inserire una resistenza di pull-up pari a 2.2 Ω in modo tale da non avere un'uscita flottante, come verrà meglio visualizzato nel *capitolo 6*.
In ingresso al comparatore di tensione sono state inserite delle resistenze di 10 kΩ e 11 kΩ per scalare il valore della tensione di alimentazione (R_{13} ed R_{14}) e dell'uscita dell'LDO (R_{11} ed R_{12}) così da non lavorare nell'intorno dei 5 V. Dopodiché viene confrontato il valore della soglia impostata a **4.75 V** (V_{in}^{+}), che verrà dimezzato per la presenza delle resistenze usate per scalare il valore della tensione di alimentazione, con l'uscita dell'LDO (V_{in}^{-}): se l'uscita dell'LDO è minore della soglia, l'uscita del comparatore è nello stato alto ($V_{in}^{+} > V_{in}^{-} \rightarrow \mathbf{1}$).

In molti casi un comparatore potrebbe rendere il sistema instabile; infatti essendo sensibile a variazioni, anche minime, del segnale intorno al valore di soglia, potrebbe originare ripetute commutazioni indesiderate ogni volta che il segnale si avvicini a tale valore, in particolare nel caso in cui al segnale utile si sovrappongano disturbi.
La configurazione di base del comparatore, infatti, può oscillare o produrre un'uscita rumorosa se l'ingresso differenziale applicato è vicino alla tensione di offset dell'ingresso del comparatore. Ciò tende a verificarsi quando la tensione sull'ingresso è uguale o molto vicina alla tensione dell'altro ingresso.
L'aggiunta di **isteresi** può evitare gli inconvenienti appena descritti. L'isteresi crea due soglie di commutazione (una per la tensione di ingresso crescente e l'altra per la tensione di ingresso decrescente).
L'isteresi è la differenza di tensione tra le due soglie di commutazione. Quando entrambi gli ingressi sono pressoché uguali, l'isteresi fa sì che un ingresso si sposti rapidamente davanti all'altro. In questo modo, l'ingresso viene spostato fuori dall'area in cui può verificarsi l'oscillazione [15].
Per il comparatore invertente utilizzato, l'isteresi viene implementata aggiungendo, alle altre due resistenze considerate per scalare il valore della tensione dell'alimentazione, una resistenza R1_hyst cosí da formare una rete di tre resistenze ed una retroazione positiva.
Ponendo $V_{T1} = 2.45$ V e $V_{T2} = 2.35$ V, che rappresentano le due tensioni di soglia, si ricava il valore della resistenza R1_hyst dalla formula presente in figura sottostante, nella quale R1_hyst equivale alla resistenza R_3.

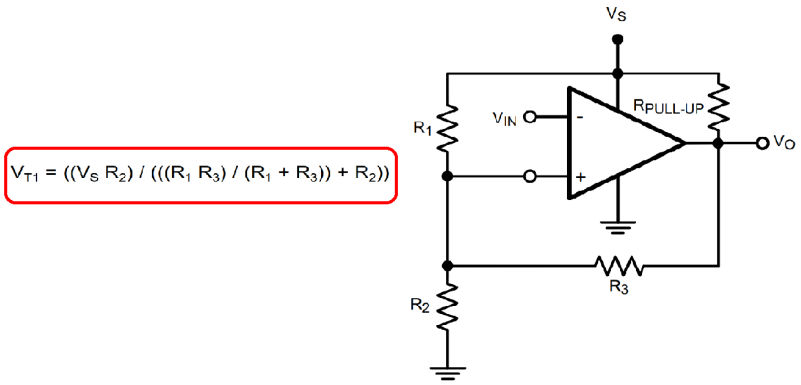


Figura 5.5: Formula per ottenere R1_hyst e schema del comparatore invertente con isteresi

Per poter applicare la formula bisogna considerare che:

- $R_1 = R_{13} = 11$ kΩ;
- $R_2 = R_{14} = 10$ kΩ;
- $V_S = 5$ V.

Si ottiene, pertanto, che: **R1_hyst** $\simeq 200$ kΩ.

In assenza di isteresi, il rumore causa attraversamenti multipli della soglia impostata, facendo scattare erroneamente il comparatore nello stato alto più volte.

5.3.4 Controllo per overvoltage (OV)

In alcuni casi la tensione in uscita dall'LDO potrebbe superare i valori nominali di funzionamento, dando cosí origine ad un incorretto funzionamento del regolatore stesso.

Per tale motivo viene realizzato un circuito per il controllo dell'overvoltage, rappresentato in figura sottostante.

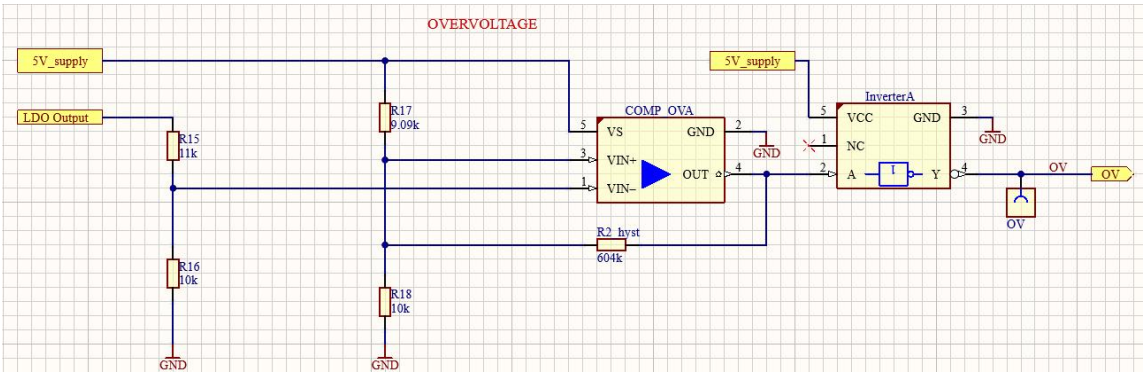


Figura 5.6: Circuito esterno all'LDO per overvoltage

Tale circuito è molto simile a quello progettato per l'undervoltage con la differenza che, rispetto al caso precedente, l'uscita del comparatore di tensione è connessa ad un **inverter** in quanto si valuta la condizione opposta rispetto a quella dell'undervoltage. Un'altra differenza è il diverso valore di una delle due resistenze connesse all'ingresso di V_{in}^{+} del comparatore poiché ciò che cambia è anche la soglia che, in tal caso, risulta essere pari a **5.25 V**, parametro che verrà dimezzato per la presenza delle resistenze usate per scalare il valore della tensione di alimentazione. Pertanto valgono le seguenti condizioni:

- se $V_{in}^{+} > V_{in}^{-} \rightarrow \quad \overline{1} = \mathbf{0}$
- se $V_{in}^{+} < V_{in}^{-} \rightarrow \quad \overline{0} = \mathbf{1}$

Anche in tal caso, come per l'undervoltage, per rendere il comparatore immune ad oscillazioni indesiderate, è stata aggiunta la resistenza R2_hyst per l'**isteresi**.

È possibile ricavare il valore della resistenza R2_hyst ponendo V_{T1} = 2.65 V e V_{T2} = 2.55 V, che rappresentano le due tensioni di soglia, e considerando che R2_hyst equivale alla resistenza R_3 nella formula presente in figura sottostante.

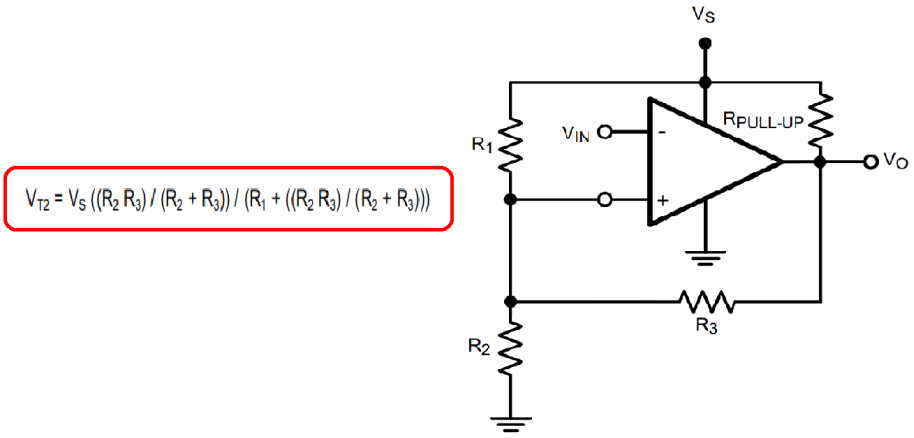


Figura 5.7: Formula per ottenere R2_hyst e schema del comparatore invertente con isteresi

Per poter applicare la formula bisogna considerare che:

- $R_1 = R_{17} = 9$ kΩ;
- $R_2 = R_{18} = 10$ kΩ;
- $V_S = 5$ V.

Si ottiene, pertanto, che: **R2_hyst** $\simeq 604$ kΩ.

5.3.5 Controllo per overcurrent (OVC)

La corrente massima di uscita erogabile dall'LDO è, secondo datasheet, 400 mA. Per questo motivo, e per evitare che a causa di carichi troppo elevati connessi la corrente superi questa soglia, viene implementata una circuiteria per il controllo dell'overcurrent, come mostrato nella figura che segue.

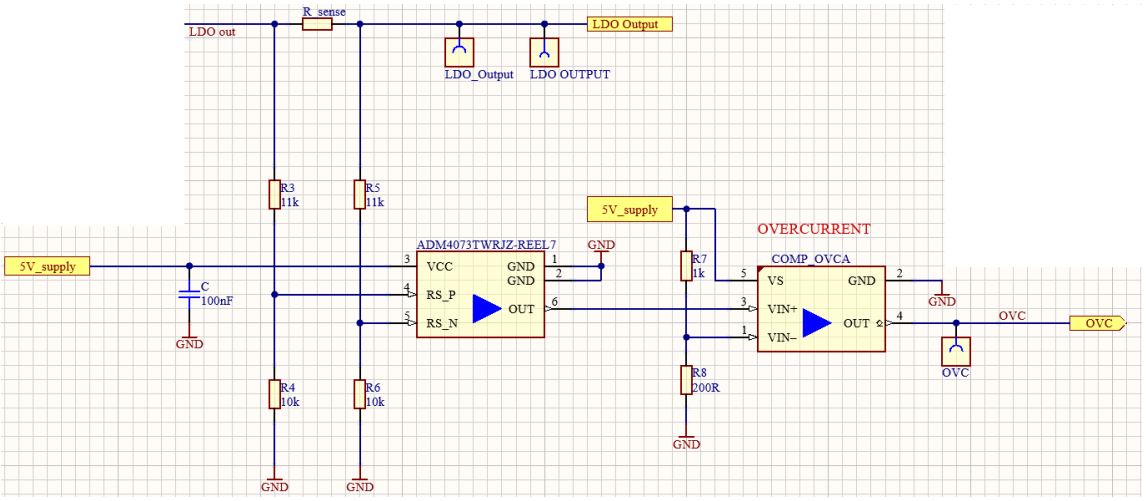
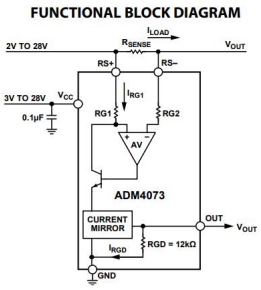
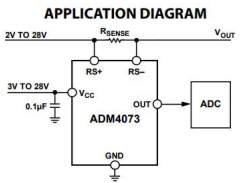


Figura 5.8: Circuito esterno all'LDO per overcurrent

Il dispositivo principale nello schema in Fig. 5.8 è l'**ADM4073TWRJZ-REEL7**, amplificatore di corrente a basso costo disponibile in tre diversi modelli di guadagno: 20 V/V, 50 V/V, 100 V/V.





Anche in tal caso sono state inserite, in ingresso all'amplificatore di corrente, delle resistenze di 10 kΩ e 11 kΩ per scalare il valore della tensione così da non lavorare nell'intorno dei 5 V.

La tensione sul pin di uscita dell'amplificatore è determinata dalla corrente che attraversa la resistenza di current-sensing e dal guadagno della versione selezionata. L'intervallo operativo va da 3 V a 28 V con una corrente di alimentazione tipica di 500 μA [16].

Si è scelto l'amplificatore con guadagno di 20 V/V ed una resistenza di sensing (**R_sense**) pari a 250 mΩ. Da questi dati si è ottenuto un valore della tensione di drop out ai capi di R_sense ed un valore della

tensione in uscita dall'amplificatore che dipendono, entrambi, dal valore della corrente in uscita dall'LDO.

La tensione in uscita dall'amplificatore andrà, poi, in ingresso al comparatore di tensione (in V_{in}^{+}) per poter essere confrontata con il valore di soglia impostato.

Le resistenze connesse al pin V_{in}^{-} del comparatore di tensione sono state opportunamente dimensionate ($R_7 = 1$ kΩ e $R_8 = 200$ Ω) per avere un valore di soglia pari a circa **0.45 V** in modo tale che, quando la corrente in uscita dall'LDO che va in ingresso alla resistenza di current-sensing supera i 400 mA, si ottiene un valore in V_{in}^{+} maggiore del valore di soglia impostato in corrispondenza del pin V_{in}^{-}.

Pertanto quando si verifica la condizione $V_{in}^{+} > V_{in}^{-}$, il comparatore produce un'uscita di un valore alto che simboleggia presenza di overcurrent.

5.3.6 Controllo per overtemperature (TS&TW)

Per ciascun regolatore integrato nel PMIC, è presente un controllo di temperatura operativa. Solitamente questa tipologia di controllo è composta da due differenti segnali generati ciascuno nell'eventualità in cui una determinata temperatura sia raggiunta dal dispositivo. Due sono le temperature considerate: a **150**° (temperature pre-warning) ed a **175**° (temperature shutdown).
Dopo una serie di valutazioni, sono state trovate le seguenti soluzioni (di cui solo l'ultima è quella effettivamente utilizzata nel progetto):

a) **Approccio iniziale**: questo è stato il primo tentativo adottato che, però, è risultato poco preciso ed efficiente.

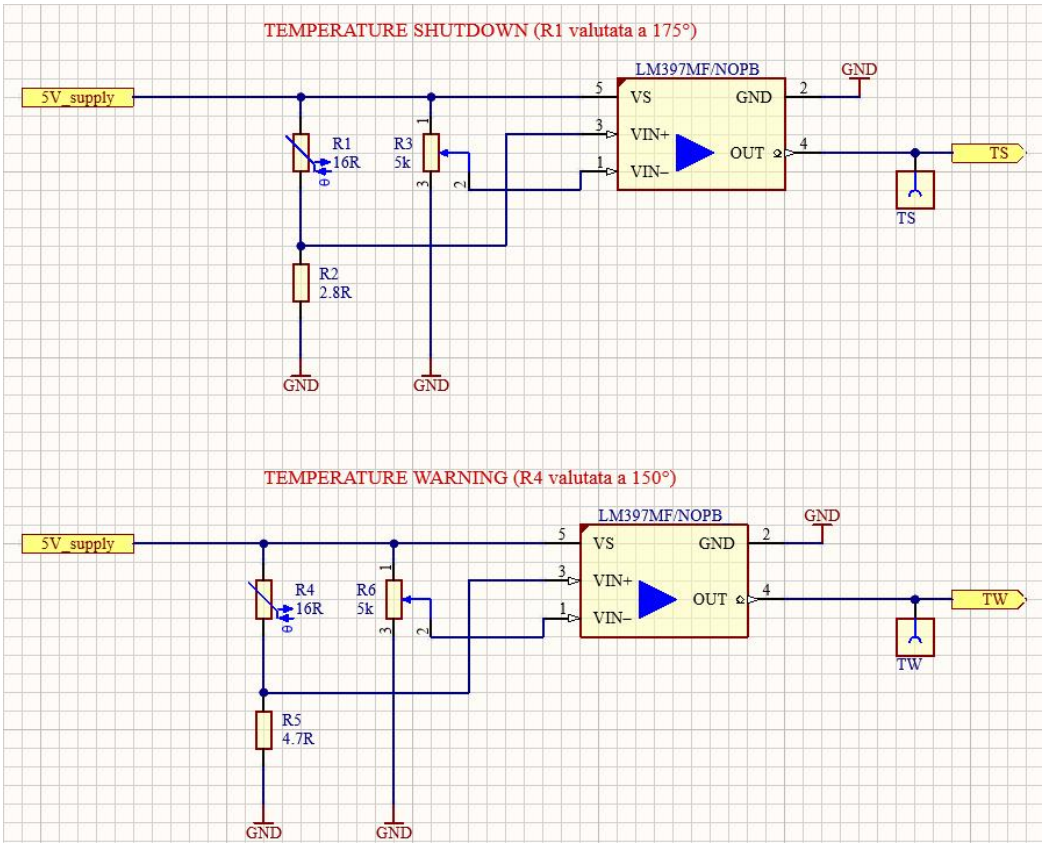


Figura 5.9: Approccio iniziale: circuito esterno all'LDO per overtemperature

Come mostrato in Fig. 5.9, l'idea è stata quella di inserire due termoresistenze di tipo NTC (Negative Temperature Coefficient), una per il TPW (temperature pre-warning) ed una per il TSD (temperature shutdown).

La termoresistenza è un sensore di temperatura che sfrutta il principio di resistenza di un metallo al variare della temperatura, nello specifico le termoresistenze di tipo **NTC** hanno la particolarità di far decrescere il valore della resistenza con l'aumentare della temperatura. Sono state posizionate in prossimità dell'LDO in modo che quando l'LDO si scalda, la termoresistenza varia il suo valore al variare della temperatura.

La termoresistenza forma, insieme ad una resistenza di tipo STD (standard), un partitore di tensione il cui valore rappresenta l'ingresso del comparatore al pin V_{in}^{+}, mentre il pin V_{in}^{-} è connesso ad un potenziometro, che rende regolabile la soglia impostata. Quando questa soglia viene superata, cioè quando si verifica la condizione $V_{in}^{+} > V_{in}^{-}$, il comparatore di tensione produce un'uscita alta segnalando overtemperature.

Tuttavia, questo metodo è poco preciso in quanto risulta difficile sia scegliere gli specifici valori delle termoresistenze che reagiscono ad un particolare valore di temperatura e sia capire quale valore di soglia impostare per il comparatore di tensione.

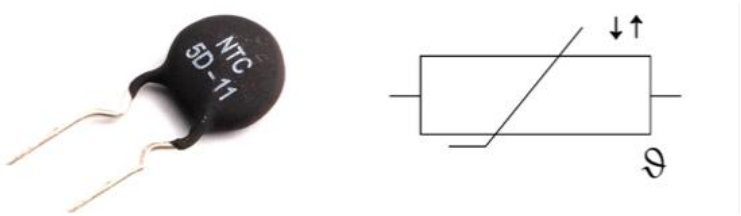


Figura 5.10: Termoresistenza di tipo NTC

b) **Approccio utilizzato**: tale approccio risulta essere quello effettivamente utilizzato per il controllo di overtemperature.

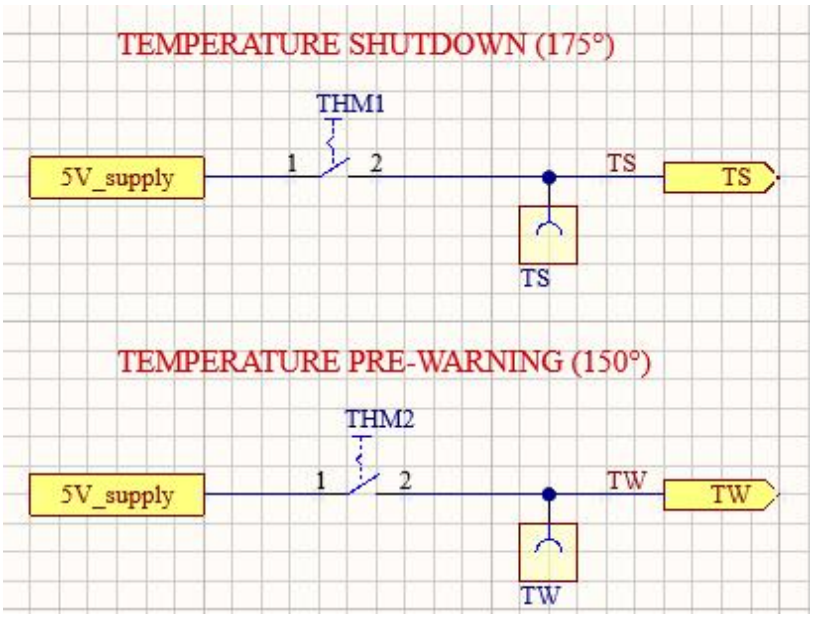


Figura 5.11: Approccio utilizzato: circuito esterno all'LDO per overtemperature

Le termoresistenze NTC sono state sostituite con due **termostati** (THM1 e THM2), uno per il temperature shutdown ed uno per il temperature pre-warning, posti in vicinanza dell'LDO.
Le caratteristiche principali sono:

- Temperatura **minima** +40°C
- Temperatura **massima** +130°C
- Temperatura di **apertura** +70°C

I termostati scelti sono termostati a disco bimetallico subminiaturizzati e ad azione di scatto positiva, che forniscono un rilevamento ed una commutazione accurata e affidabile in un unico dispositivo. Anch'essi, come già visto per le termoresistenze, sono stati posizionati in prossimità dell'LDO: quando l'LDO si scalda, la striscia bimetallica dei termostati agisce come un interruttore di contatto elettrico in un circuito di

riscaldamento elettrico e, quando viene raggiunta la temperatura desiderata, il circuito è interrotto ed i termostati producono un'uscita alta segnalando così presenza di overtemperature.

Tali componenti possono essere posizionati e saldati automaticamente su schede PCB con apparecchiature automatiche ad alta velocità, eliminando la necessità di una costosa manodopera per il posizionamento, che oggi è richiesto per la maggior parte dei termostati di alimentazione.

Questi termostati hanno una capacità di commutazione fino a 0.5 A per 48 VDC e raggiungono una commutazione a basso livello fino a 0.001 A-0.020 A a 5 VDC per 100.000 cicli. La temperatura è preimpostata in fabbrica e non è regolabile sul campo. I campi di applicazione includono computer e periferiche, aerei, automobili, dispositivi medici e apparecchiature di test [17].

Il grande **vantaggio** di usare i termostati è che essi, rispetto alle termoresistenze, producono direttamente un'uscita digitale e questo porta ad un'assenza del comparatore di tensione LM397MF/NOPB.

Nonostante le condizioni di temperatura definite inizialmente fossero pari a 150° per il temperature pre-warning ed a 175° per il temperature shutdown, per gli scopi di tale tesi e del prototipo di interfaccia con l'FPGA è stato sufficiente generare un solo segnale, attraverso uno stimolo esterno, che fosse in grado di raggiungere una temperatura operativa minore di quella prefissata all'inizio e cioè pari a **70**°.

5.4 Presenza del microcontrollore XMC 2Go

L'ultimo step per la parte schematica della PCB prevede la necessità di inserire un **microcontrollore**. Recentemente Infineon ha introdotto sul mercato una piccola scheda di sviluppo con a bordo un microcontrollore (XMC1100) ed un altro circuito integrato che fa da programmatore/debugger: la XMC 2Go. Quest'ultimo è progettato per valutare le capacità del microcontrollore XMC1100 e la potente catena di tool gratuiti come DAVE.
DAVE è l'ambiente di programmazione di Infineon che offre un'interfaccia grafica sia per definire i componenti (in DAVE sono chiamati "applicazioni") utilizzati per il proprio programma e sia per "tradurli" in librerie C che saranno poi impiegati nella compilazione del codice[18].
Il kit include 64 kB di memoria flash, 16 kB di RAM, oscillatore interno, due basette a 8 pin, debugger J-Link incorporato, USIC a 2 canali, ADC a 6 canali, 4 timer a 16-bit, interrupt esterni, clock in tempo reale, generatore numerico casuale e 2 LED utente [19].
L'aggiunga dell'XMC 2Go è nata da alcune esigenze del Digital Team di Infineon: poter agire sulle porte di OV, UV, TS, TW e OVC in modo digitale, senza l'ausilio della board. Con il microcontrollore i colleghi del Digital Team sono in grado di agire sull'FPGA, permettendo la generazione di segnali, senza dover obbligatoriamente far funzionare l'LDO.
Quindi, a livello digitale, è possibile fare verifiche sulle funzionalità dell'FPGA, generando segnali con l'XMC, anche senza utilizzare la board realizzata.
La board dell'XMC 2Go si presenta con 16 pin, dei quali, a parte i due utilizzati per l'alimentazione, tutti gli altri sono porte di tipo GPIO.
Di tutti i pin si ha che: un ristretto numero non accessibile viene utilizzato per la comunicazione con il debugger, i pin 2.1 e 2.2 sono utilizzati per comunicare con il PC attraverso una seriale virtuale (SPI) mentre i pin 1.0 e 1.1 sono connessi a due

LED indicatori per offrire un minimo di interazione con il programmatore.
Tra i pin GPIO, ne sono stati selezionati cinque da connettere alle porte TS (temperature shutdown), TW (temperature pre-warning), UV (undervoltage), OV (overvoltage) e OVC (overcurrent).
Nella figura sottostante viene mostrata la 'Virtual Pin View' dell'XMC1100 su DAVE e la relativa 'Pin Header'.

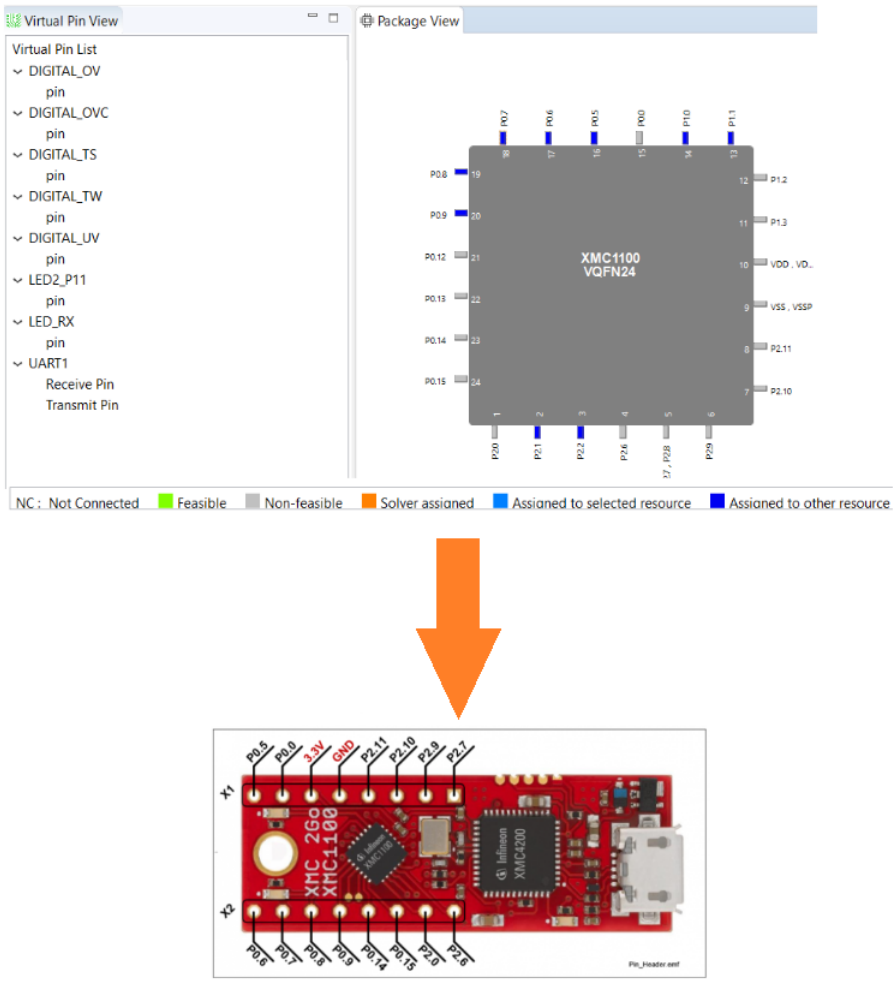


Figura 5.12: 'Virtual Pin View' dell'XMC1100 su DAVE e la 'Pin Header' dell'XMC 2Go

L'assegnazione dei pin è la seguente: **P0.5** → **OVC**; **P0.6** → **TS**; **P0.7** → **TW**; **P0.8** → **UV**; **P0.9** → **OV**.

5.5 Altium PCB Board Layout

Dopo aver realizzato tutti gli schematici, l'ultimo step per la realizzazione della PCB prevede il layout.

La fase di progettazione della PCB comprende l'impostazione del design tool, il profilo della scheda, l'importazione della netlist, il posizionamento dei componenti, il routing, la pulizia della serigrafia, il controllo DRC (Design Rule Checking) e la generazione dei documenti per la produzione della PCB (gerber, netlist, ecc.).

Si procede con la creazione di un file .PcbDoc sul quale vengono importati tutti i vari schematici, poi si posizionano correttamente i componenti ed infine si esegue il routing. Il routing è la creazione di piste per l'effettiva connessione dei vari componenti seguendo le regole definite dallo schematico. Nel caso della board progettata per tale tesi, il routing viene effettuato su due piani: top e bottom, consentendo in questa maniera di evitare difficoltá nella connessione dei vari componenti causata dall'intersezione di linee.

Dopo aver terminato il routing bisogna creare dei piani di massa (poligoni di ground) cosí da non avere la necessità di collegare ogni componente al ground ma, attraverso questo poligono di ground, si avrà automaticamente ogni pad collegato al piano di massa.

Figura 5.13: Un esempio di piano di massa in Altium Designer

Come già accennato in precedenza, per questa board sono stati utilizzati **due** layer: un layer superiore (top layer) ed uno inferiore (bottom layer) pertanto è stato necessario realizzare due piani di massa per entrambi i layer.

Dopo aver verificato che non ci sia la presenza di 'violations' ed errori, l'ultima azione da eseguire è il DRC: se il DRC non da errori, la PCB può essere considerata terminata.

Nelle figure sottostanti sono riportate le rappresentazioni in **2D** del top layer e bottom layer.

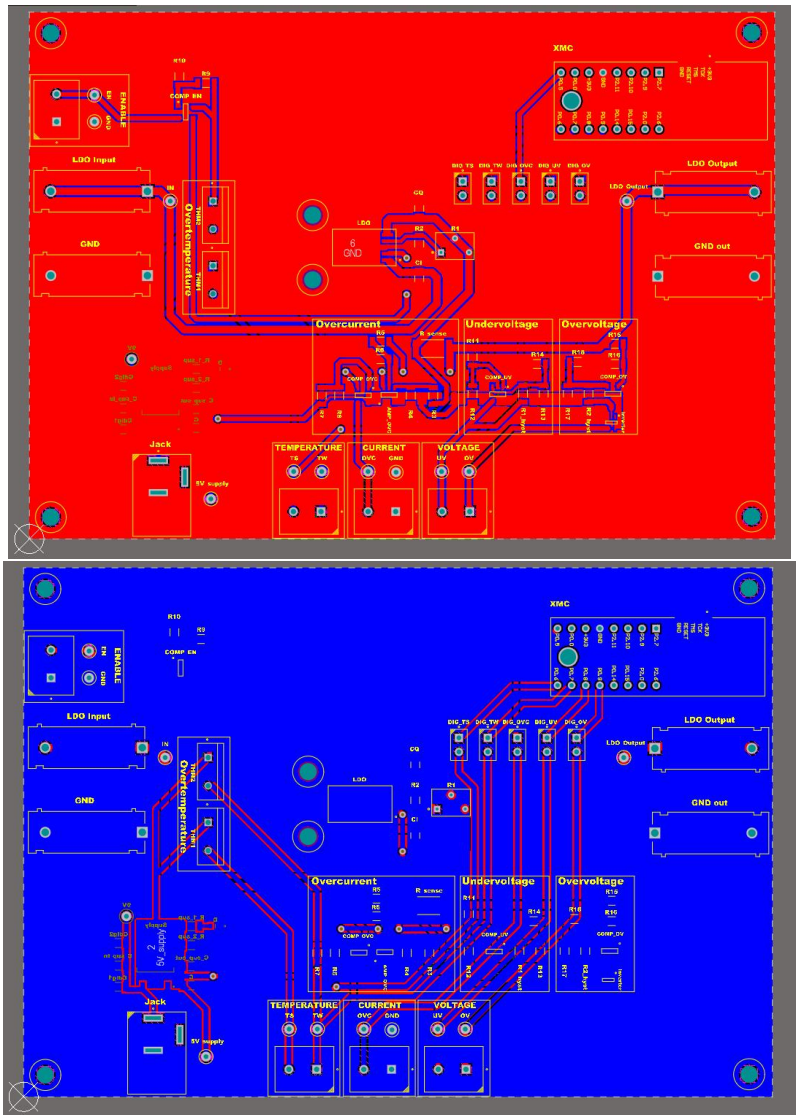


Figura 5.14: Top layer e bottom layer in 2D

In Altium è anche possibile visualizzare la rappresentazione in 3D della board, come mostrato nelle figure 5.15 e 5.16.

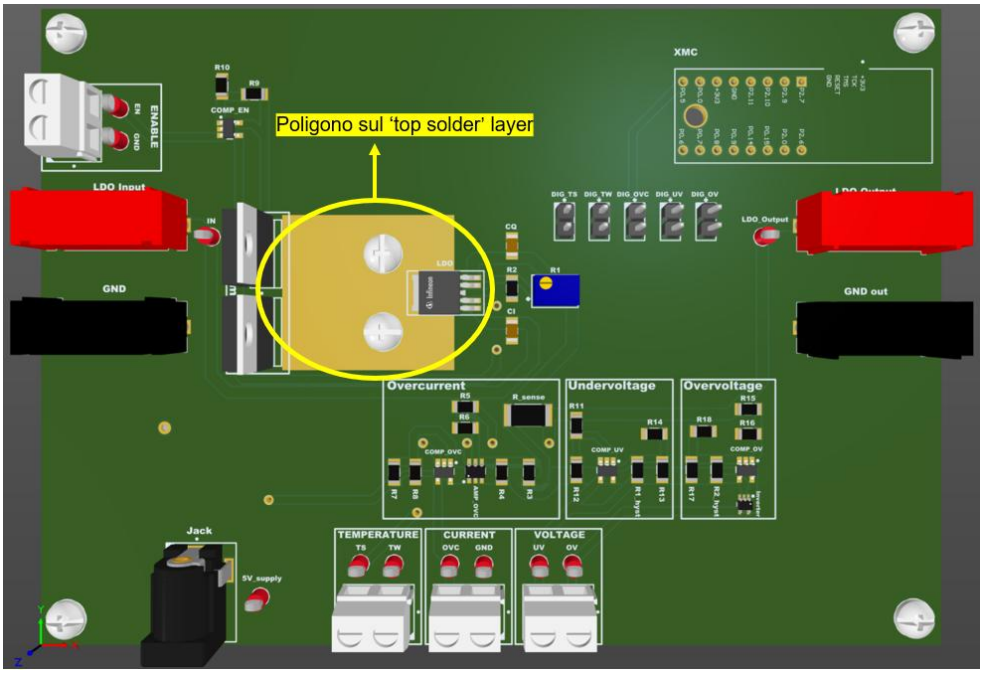


Figura 5.15: Top layer in 3D

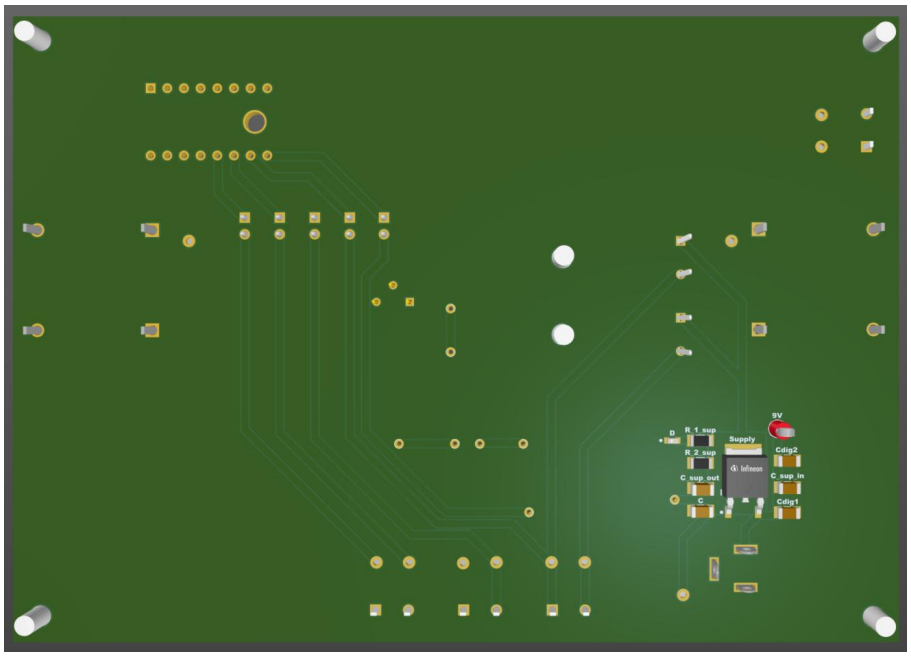


Figura 5.16: Bottom layer in 3D

Come evidenziato in Fig. 5.15, sul layer superiore è presente un ulteriore poligono di ground inserito sul layer 'Top Solder', sul quale è presente sia l'LDO sia i due termostati (che verranno piegati su tale poligono) per poter eseguire i vari test per il controllo di overtemperature; in tal modo si eviteranno misure errate di temperatura e dissipazione di calore.

Sul bottom layer, invece, sono stati saldati solo i componenti utilizzati per fornire alimentazione, come spiegato nella sezione 5.2.

Infine, sono stati generati tutti i vari documenti necessari per poter effettuare l'ordine della board sul sito 'Eurocircuits'.

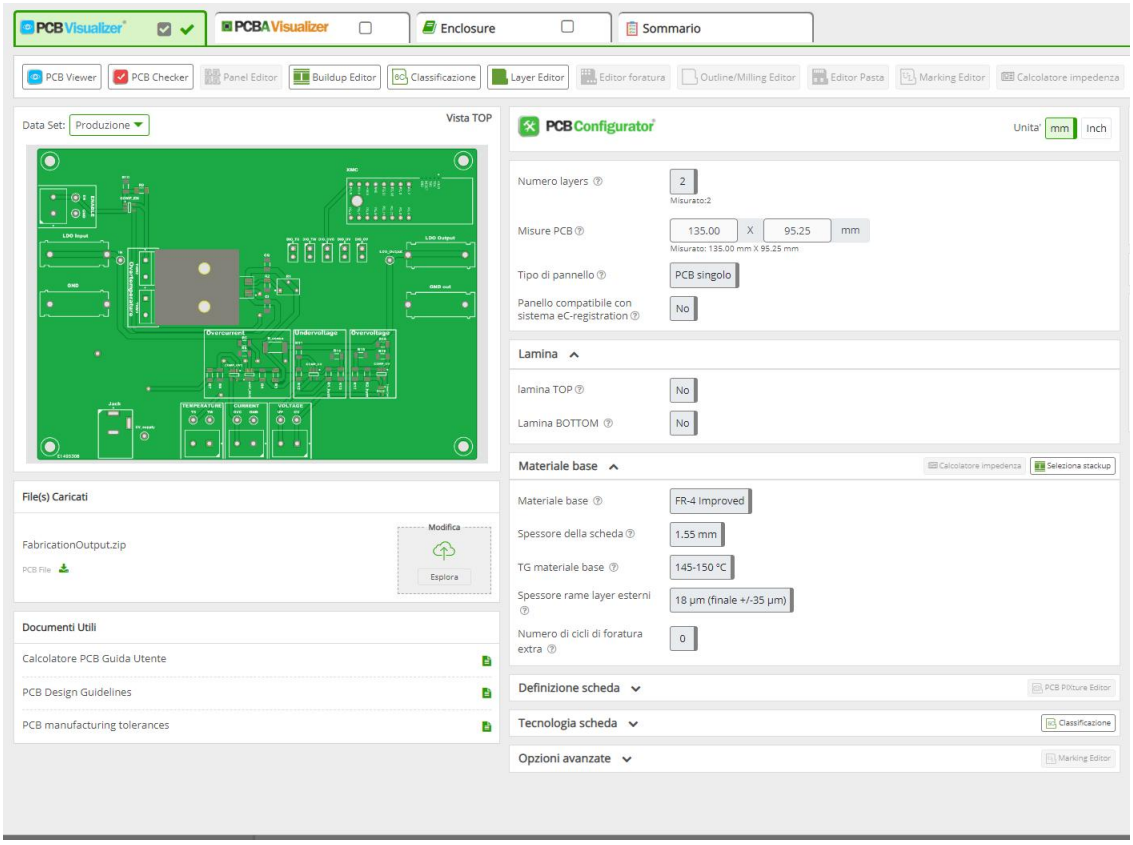


Figura 5.17: Panoramica del sito Eurocircuits per ordine PCB

La board ordinata non prevedeva la saldatura dei componenti che, invece, sono stati saldati manualmente presso il laboratorio di Infineon Technologies.

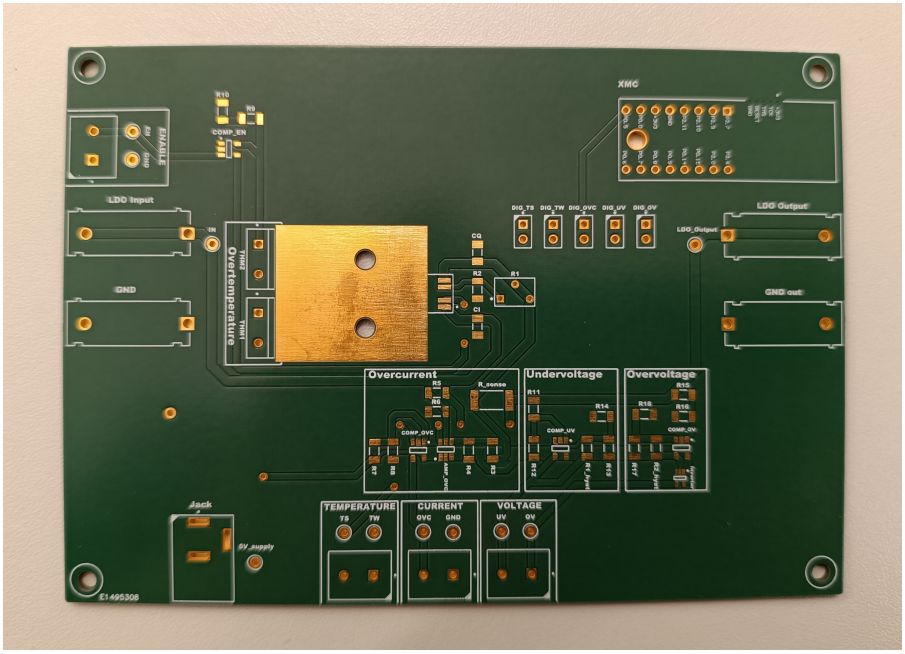


Figura 5.18: Top layer PCB

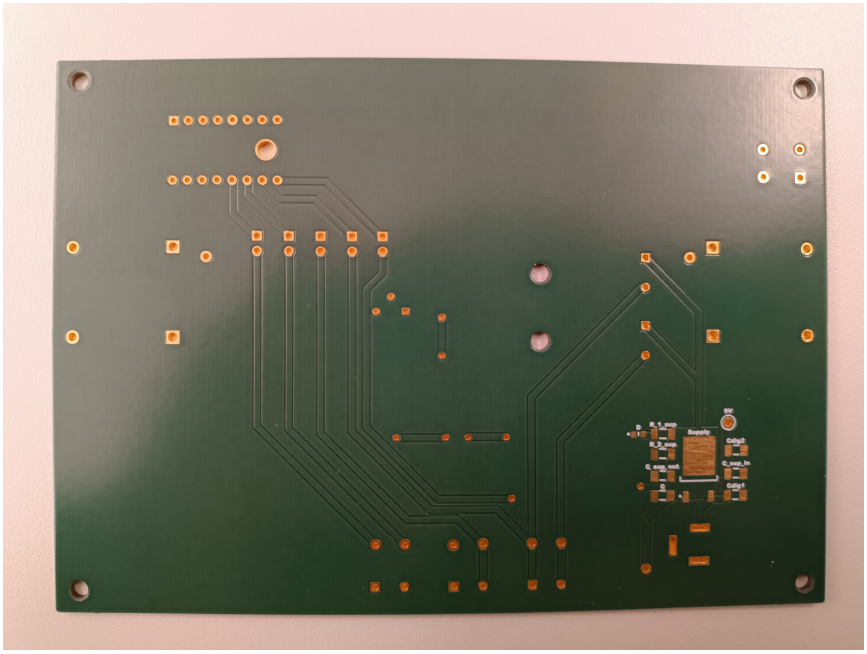


Figura 5.19: Bottom layer PCB

Capitolo 6

Test in laboratorio e risultati

6.1 Collaudo scheda

Dopo aver saldato i componenti sulla board, si passa a verificare il suo corretto funzionamento tramite specifici test operativi.
La figura 6.1 mostra la PCB con tutti i componenti saldati correttamente.

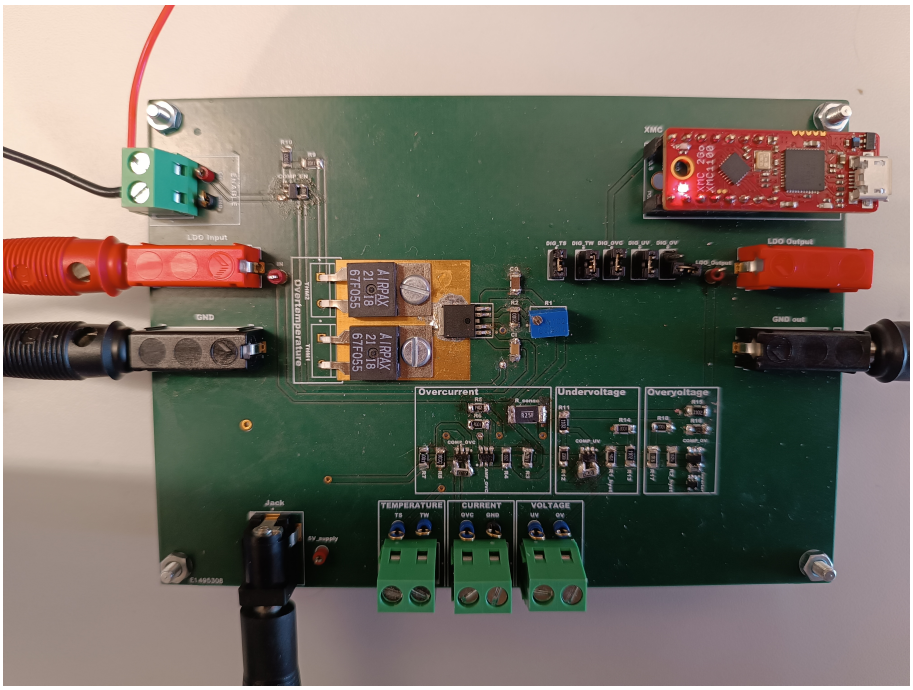


Figura 6.1: PCB con componenti saldati

Prima di effettuare i vari test, è necessario accertarsi che l'LDO selezionato regoli effettivamente a 5 V come da datasheet e che i vari componenti funzionino come previsto. Per fare ciò si procede collegando i vari strumenti descritti nel paragrafo 3.4 alla board:

- Generatore di tensione connesso al pin di input della board: permette l'accensione del regolatore lineare di tensione fornendogli 12 V in ingresso;
- Sourcemeter connesso all'enable dell'LDO: vengono forniti 5V all'enable del regolatore lineare di tensione per abilitare l'LDO al funzionamento;
- Multimetro connesso all'uscita dell'LDO per poter misurare i valori di tensione forniti in uscita dal dispositivo: dalle figure sottostanti è possibile notare che il valore in uscita dal regolatore lineare di tensione, pari a circa 5 V, è proprio quello che ci si auspicava di ottenere.

Come mostrato nella figura che segue, ci si posiziona in corrispondenza del testpoint 'LDO_Output' con un puntale connesso all'ingresso del multimetro.

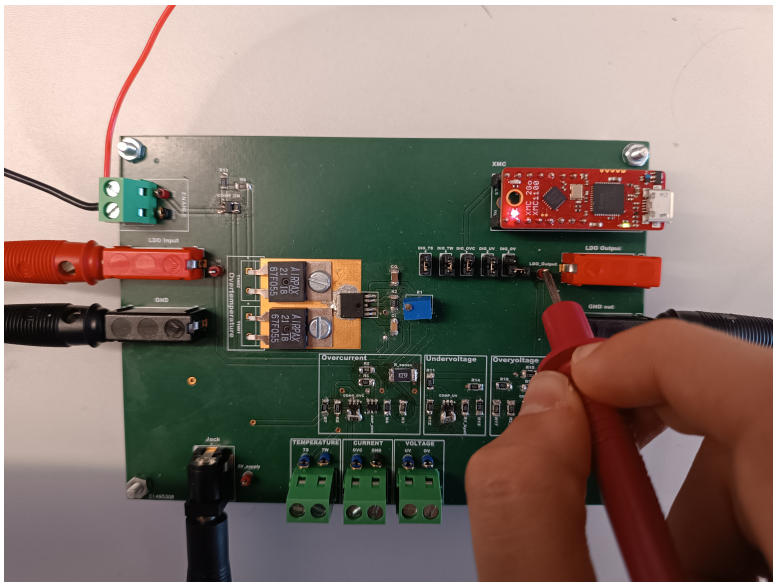


Figura 6.2: Misurazione della tensione in uscita dall'LDO

Dopo aver posizionato il puntale e connesso la PCB a tutti gli strumenti descritti precedentemente, si ottiene il valore della tensione in uscita dall'LDO come mostrato in figura sottostante.

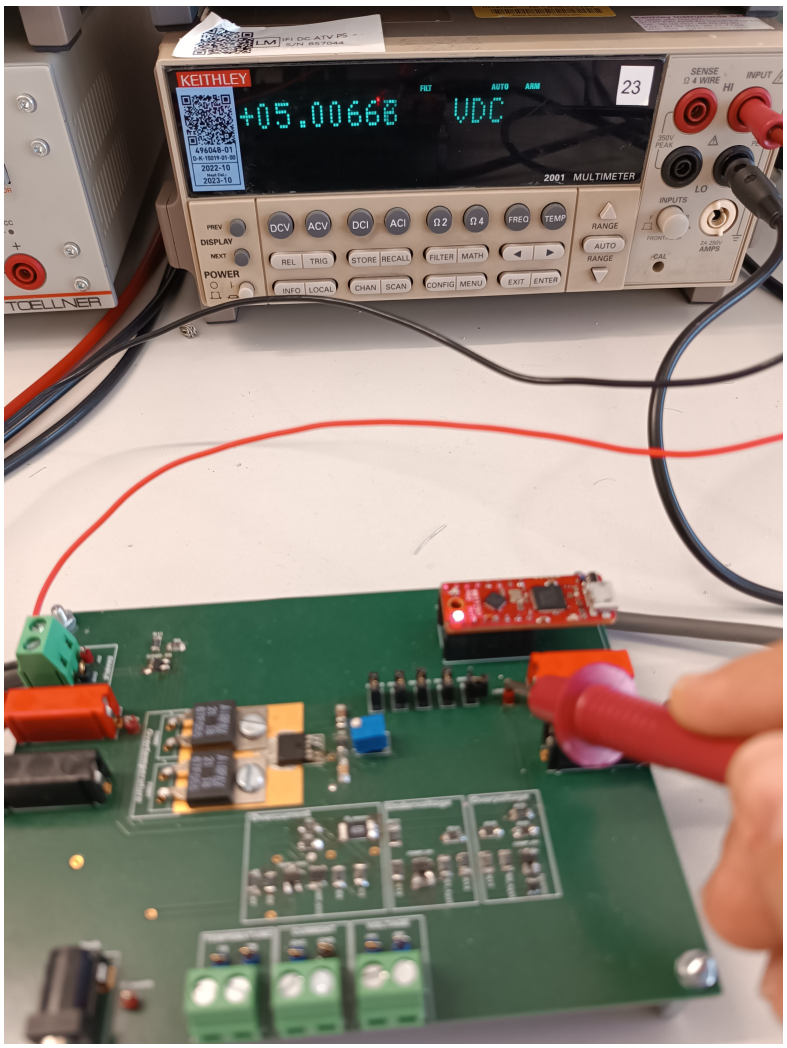


Figura 6.3: Valore della tensione in uscita dell'LDO

Dopo aver testato la corretta regolazione dell'LDO, il passo successivo è stato verificare, prima di procedere con i vari test, l'effettivo funzionamento dei comparatori di tensione per i controlli di UV, OV, OVC, TS, TW (compreso quello relativo al segnale di enable).

L'inserimento di una resistenza di pull-up in serie ad ogni comparatore di tensione si è resa necessaria a causa della sua configurazione di tipo 'open collector', grazie alla

quale si evita di avere un'uscita flottante. Con tale resistenza, infatti, è stato possibile verificare che i vari comparatori di tensione producono un'uscita alta o bassa a seconda del confronto tra il valore di riferimento ed il valore della tensione che arriva in uscita dall'LDO, come descritto nel capitolo 5.3.

6.2 Modello analogico dell'LDO

Per testare un modello fisico di sistema si è deciso di utilizzare la scheda DUT (Device Under Test) che simula le funzioni di un LDO, la cui progettazione è descritta nel capitolo 5. Questa scheda DUT è basata su un regolatore LDO TLE42764 attualmente prodotto da Infineon.

Tale LDO prende 9 V in ingresso e li regola ad una tensione di 5 V, che viene inviata in ingresso ai vari componenti per alimentarli. Questi componenti rilevano le anomalie di funzionamento mediante l'uso di comparatori di tensione e con un segnale logico in uscita le comunicano all'FPGA, quest'ultima opportunamente connessa alla board tramite i connettori inclusi. Tale board include anche un connettore per il segnale di enable, che è a sua volta collegato all'FPGA e ne consente l'abilitazione tramite gli I/O dell'FPGA.

Se la si collegasse direttamente all'FPGA, i 5 V provenienti dai segnali probabilmente danneggerebbero il chip in quanto, secondo le linee guida di Xilinx, il chip supporta un ingresso massimo assoluto di 4 V.

In questo caso, tuttavia, la scheda PXI contiene circuiti di protezione che permettono di sopportare fino a 5.5 V senza danneggiare il chip. Ciò evita l'aggiunta di qualsiasi circuito aggiuntivo per collegare i sistemi.

Non si può dire lo stesso per i segnali che vanno dall'FPGA all'LDO. La scheda può emettere solo segnali con tensioni massime di 3.3 V dai suoi pin di I/O digitali, mentre la tensione minima che l'interruttore di enable dell'LDO riconosce come segnale alto è di un valore di 3.5 V. Questo disallineamento provoca una situazione in cui l'FPGA non è in grado di abilitare da solo l'LDO pertanto, come descritto meglio nella sezione 5.3.1, viene aggiunto un comparatore di tensione per risolvere tale problema.

In questo modo, l'FPGA è in grado di attivare e disattivare l'LDO a suo piacimento. Alla scheda contenente l'LDO è stato aggiunto un microcontrollore per emulare i segnali di uscita dell'LDO senza doverlo alimentare. Ciò consente di eseguire test in cui le funzioni sono controllate programmaticamente, invece di essere attivate da condizioni operative particolari in cui si porta il circuito.

La figura sottostante mostra la PCB insieme al microcontrollore, utilizzato per il collegamento SPI, ed al connettore del pannello frontale della scheda PXI che, tramite una scheda di I/O, consente di inviare e ricevere segnali tra il modello e l'FPGA.

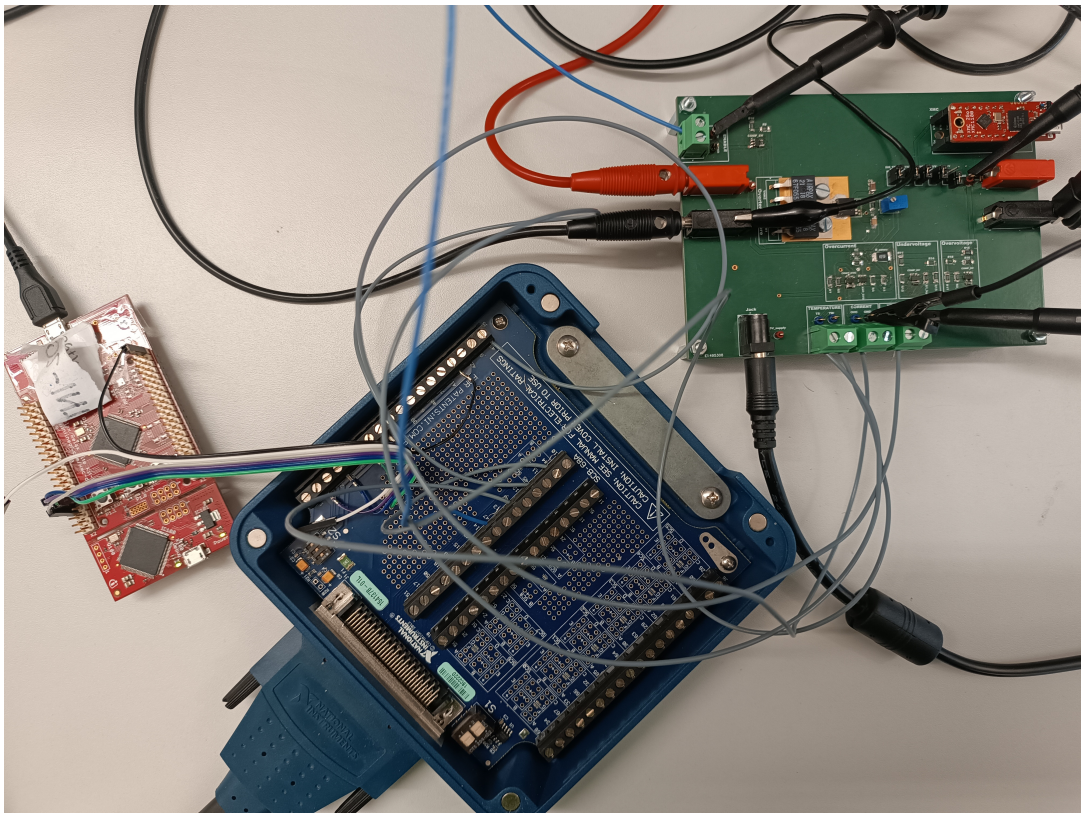


Figura 6.4: Connessione tra PCB-microcontrollore-FPGA

6.3 Circuito di trigger dell'oscilloscopio

Il trigger è un circuito elettronico presente in tutti gli oscilloscopi ed ha la funzione di sincronizzare la partenza della scansione orizzontale con un preciso livello di soglia del segnale periodico da analizzare. La sincronizzazione permette la visualizzazione stabile sullo schermo del segnale in analisi.
Due sono le modalità di trigger che vengono utilizzate per questi test:

- **Normal**: la scansione ricomincia solo in presenza dell'evento di trigger. Al termine della scansione la traccia viene riportata al punto iniziale (a sinistra dello schermo) e resta in attesa del prossimo evento di trigger: in tale modalità, se mancano gli eventi di trigger, non si vede alcuna traccia;

- **Single**: la scansione avviene solo una volta, al primo evento di trigger. Per attenderne un'altra si deve ri-abilitare manualmente il trigger con l'apposito pulsante: in questo modo si vede solamente un'unica traccia al primo evento di trigger la cui durata dipende dall'impostazione dell'asse dei tempi.

Considerando che il circuito di trigger di un oscilloscopio ha lo scopo di far partire la memorizzazione (e quindi la visualizzazione) del segnale di ingresso all'istante desiderato, l'opportuna regolazione di tale circuito permette di visualizzare in modo stabile un segnale ripetitivo oppure di catturare la parte voluta di un segnale non ripetitivo. In questa modalità la traccia sullo schermo dell'oscilloscopio inizia quando il segnale in ingresso supera un determinato valore.
È inoltre possibile anche selezionare la pendenza con la quale il segnale supera il valore di soglia di trigger: **pendenza positiva** se la soglia viene superata mentre il segnale sta crescendo oppure **pendenza negativa** se la soglia viene superata mentre il segnale sta decrescendo.

6.4 Test per il controllo di UV

Per l'acquisizione delle forme d'onda relative ai vari test, si utilizza l'oscilloscopio Lecroy HDO6054.

Per una buona acquisizione della traccia bisogna impostare la corretta scala orizzontale e verticale sull'oscilloscopio.

Figura 6.5: Pannello frontale dell'oscilloscopio

Dal pannello frontale mostrato nella figura precedente, si ruota la manopola per impostare sull'asse delle ascisse il tempo/divisione (secondi) e sull'asse delle ordinate il guadagno verticale (Volt/divisione).

In seguito si setta la modalità di trigger, che ha la funzione di sincronizzare la partenza della scansione orizzontale con un preciso livello di soglia del segnale periodico da analizzare.

Successivamente si procede con la verifica per il controllo di undervoltage: si confrontano l'uscita dell'LDO e la forma d'onda del segnale di enable con il segnale di undervoltage.

Si connettono tre sonde all'oscilloscopio per visualizzare tre diverse tracce: il segnale in uscita dall'LDO, il segnale di enable ed il segnale di undervoltage.

Come mostrato in fig. 6.4, alcuni componenti della board vengono connessi all'FPGA tramite la scheda di I/O.

Mediante la scheda di I/O, il cui schema è riportato in figura sottostante, è possibile pilotare i segnali di EN, UV, OVC, TS e TW tramite l'FPGA.

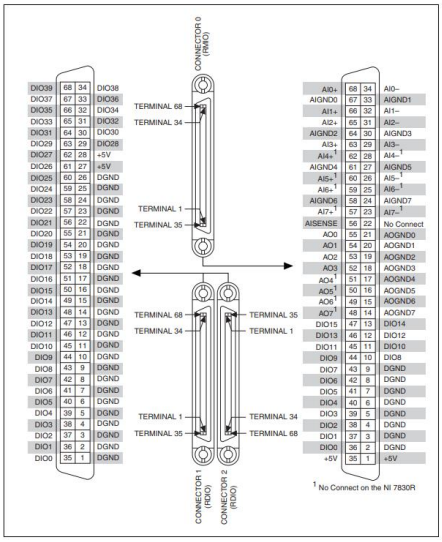


Figura 6.6: Assegnazioni e posizioni dei pin del connettore

Nel caso dell'UV si considera:

- Pin di EN connesso al DIO12;
- Pin di UV connesso al DIO8.

La parte digitale relativa all'FPGA è stata, invece, curata da un altro tesista di Infineon Technologies che, nello specifico, si è occupato del processo di porting del progetto sull'FPGA ed in seguito della creazione di blocchi che gestiscono il funzionamento di LDO emulati e/o reali mediante un sistema LabVIEW-FPGA, in modo tale da rendere possibile la comunicazione con la PCB.

Tuttavia, nonostante l'FPGA rappresenti la logica di controllo dell'LDO, non costituisce tutta la logica che controlla il PMIC.

È stato necessario creare, su LabVIEW, una batteria virtuale sia perchè alcuni blocchi hanno bisogno di un valore di tensione di riferimento per funzionare e sia perchè alcuni segnali dipendono da questa tensione di alimentazione per cambiare stato.

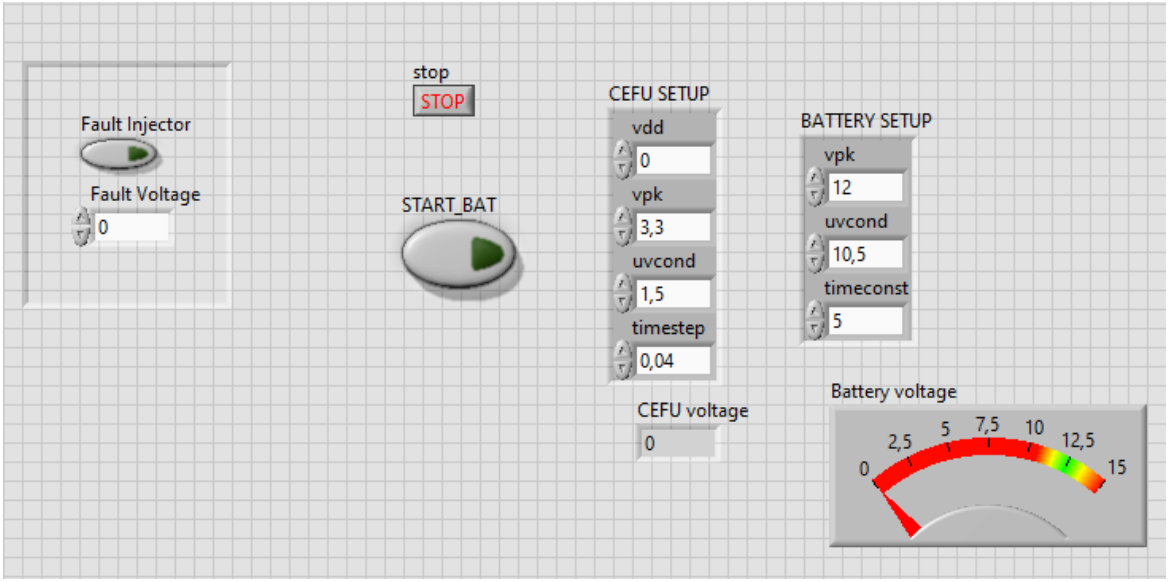


Figura 6.7: Modello di una batteria virtuale creata su LabVIEW

I **registri** all'interno dell'FPGA sono monitorati da un computer separato rispetto a quello in cui sono stati realizzati i modelli LabVIEW VI, tale computer è collegato all'FPGA tramite l'interaccia SPI di un microcontrollore (XMC Infineon) che fornisce un ponte tra i protocolli USB-UART e SPI.

Per accelerare lo sviluppo, è stata utilizzata una evaluation board Infineon XMC con un firmware e un software di controllo sviluppati internamente.

Questo software è già utilizzato per eseguire la validazione dei componenti nei laboratori e la sua funzione è quella di scrivere sui registri di configurazione all'interno dei dispositivi e di leggere i registri di stato del sistema da testare.

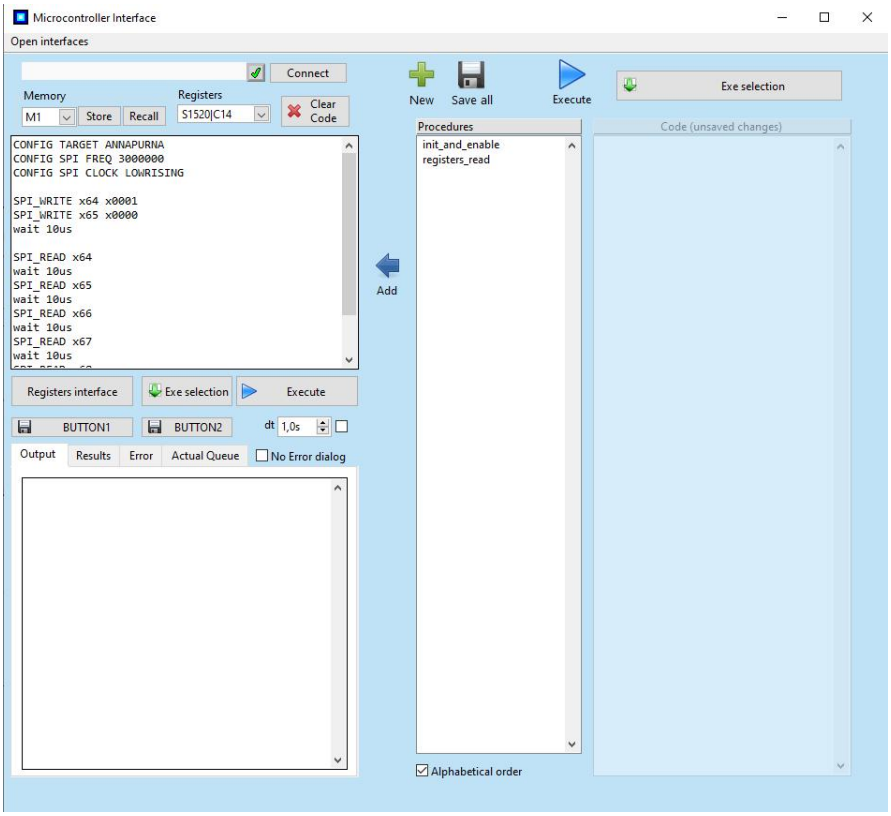


Figura 6.8: Interfaccia per il microcontrollore di Infineon

Dopo aver considerato tutti questi processi, si procede con la verifica per il controllo dell'undervoltage.

I passi da seguire sono i seguenti:

- Connettere tre sonde all'oscilloscopio considerando:
  - il canale 1 connesso all'**enable**;
  - il canale 3 connesso all'**uscita** dell'**LDO**;
  - il canale 4 connesso al segnale di **UV**;
- Impostare il **trigger** in modalità 'normal' sul canale 1;
- Abilitare la batteria virtuale premendo sul tasto 'START_BAT' (vedi fig. 6.7). Essa funge da sourcemeter, ma a livello digitale, ed è necessaria per abilitare l'enable fornendogli una certa tensione in ingresso;
- Abilitare l'**enable** tramite l'interfaccia per il microcontrollore di Infineon (fig. 6.8), cliccando sul codice 'init_and_enable' → 'Execute';
- Eseguire un controllo di lettura sui **registri** (cliccare sul codice 'register_and_read' → 'Execute') e capire a quale azione di risposta corrispondono.

In tal modo è possibile interrogare l'FPGA, che risponderà agli stimoli inviati dalla board.
In particolare, in questo primo test per l'UV si verifica anche la fase di 'SOFT START'. La funzione di '**SOFT START**' permette di non avere brusche partenze del dispositivo e conseguenti elevati spunti di corrente: è una tempistica di qualche ms che permette di avviare un dispositivo in modo graduale generando una **rampa** e non un gradino. Tale fase ha inizio con l'accensione del dispositivo e la relativa abilitazione dell'enable: non appena il segnale di enable diventa alto, inizia la fase di 'SOFT START' dell'LDO che si accende in modo graduale generando una rampa.
Nella figura che segue è possibile evidenziare il segnale di enable, il segnale di uscita dell'LDO ed il segnale di undervoltage: si nota che, non appena l'enable viene abilitato (segnale in giallo), l'LDO inizia a regolare fino ad assestarsi al valore di circa 5 V (segnale in blu); il segnale di UV (segnale in verde), invece, risulta essere alto fin quando l'LDO non raggiunge il valore di uscita pari a 5 V.

Infatti, non appena l'LDO termina la fase di 'SOFT START', il segnale di UV diventa basso.

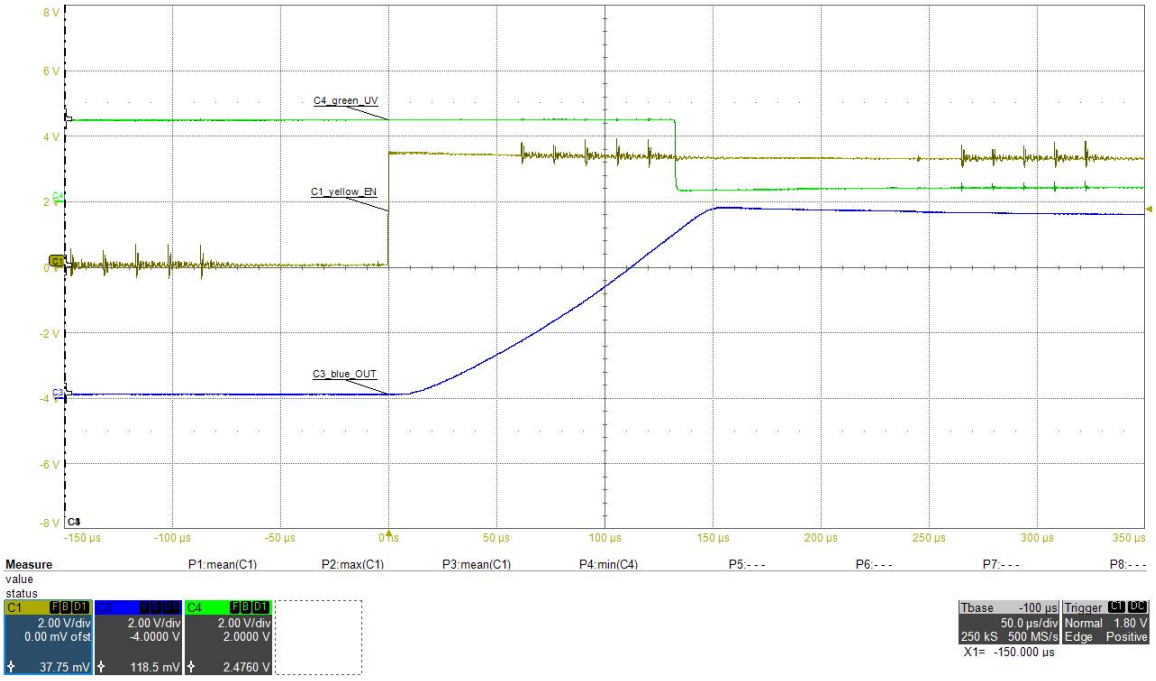


Figura 6.9: Controllo di UV

Infine, bisogna eseguire il controllo di lettura dei registri dell'FPGA: in particolare, quando si segnala presenza di UV, si dovrebbe verificare la condizione di modifica del bit (da bit 0 a bit 1) del registro che corrisponde all'UV.

Dallo studio della parte digitale effettuata dall'altro tesista, è emerso che dovrebbe cambiare il bit in posizione 1 del registro 0x68.

Dallo schema ottenuto dall'interfaccia per il microcontrollore di Infineon risulta, infatti, che il bit in posizione 1 del registro 0x68 è alto segnalando, pertanto, presenza di UV.

```
Execution ended
SPI_READ X64 --> 0000000000100001 (
01011101010000000000000000100001 )
WAIT 10 --> OK
SPI_READ X65 --> 0000000000000000 (
11100010010000000000000000000000 )
WAIT 10 --> OK
SPI_READ X66 --> 0000000000000000 (
11100010010000000000000000000000 )
WAIT 10 --> OK
SPI_READ X67 --> 0000000000000010 (
11100001010000000000000000000010 )
WAIT 10 --> OK
SPI_READ X68 --> 0000000000000010 (
11100001010000000000000000000010 )
WAIT 10 --> OK
SPI_READ X69 --> 0000000000000000 (
11100010010000000000000000000000 )
WAIT 10 --> OK
```

Figura 6.10: Lettura registri dell'FPGA per segnale di UV

Interrogando l'**FPGA**, quindi, è possibile effettuare test in cui le funzioni sono attivate dal modello virtuale mentre, grazie alla **PCB**, si ottiene un modello fisico che riesce a simulare le varie funzioni dell'LDO.

6.5 Test per il controllo di OVC

L'LDO utilizzato per questa tesi prevede un range di corrente massimo di 400 mA. Come carico di corrente in laboratorio si possono utilizzare due approcci:

- collegando un carico elettronico all'LDO;
- collegando un carico **passivo** come quello mostrato in figura sottostante.

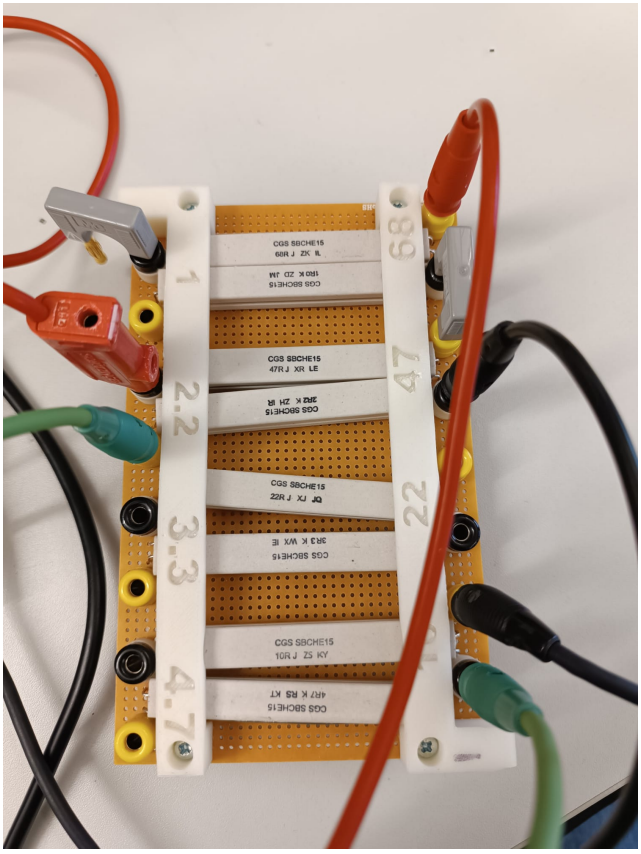


Figura 6.11: Carico passivo pari a 12.2 Ω

Per avere il valore massimo di corrente di 400 mA in uscita dall'LDO, si considera il carico passivo di un valore pari a circa 12.2 Ω.

Il circuito che rappresenta il carico di corrente realizzato in laboratorio ed utilizzato per i test di OVC è rappresentato in Fig. 6.12.

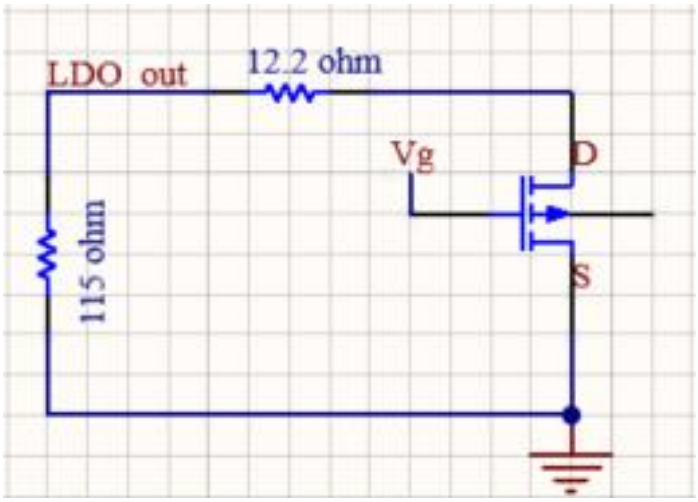


Figura 6.12: Schematico del circuito realizzato in laboratorio per effettuare test di OVC

Si noti che il carico passivo è pilotato da un mosfet, il quale presenta una tensione di accensione V_{GS} pari a 4V.

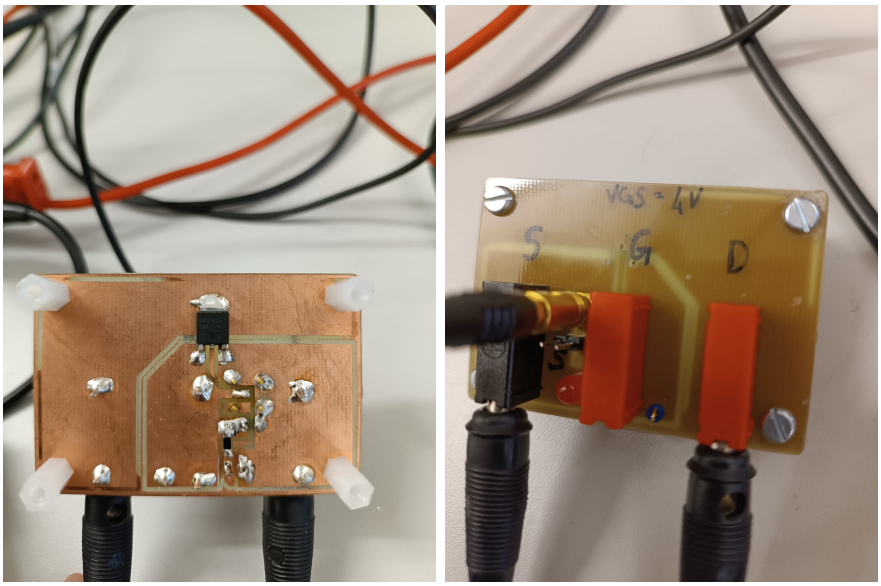


Figura 6.13: Mosfet con tensione di accensione V_{GS}=4V

Per effettuare test per il controllo di OVC considero il waveform generator, descritto nel capitolo 3.4, che viene utilizzato per pilotare il gate del mosfet.
Pertanto si ha il carico passivo di 12.2 Ω pilotato dal gate del mosfet che, a sua volta, è anche connesso al generatore di forme d'onda che genera un impulso. Il pannello frontale del generatore di forme d'onda è impostato con i seguenti parametri:

- Forma d'onda: impulso;
- Periodo: 10 s;
- Conteggio burst: 1 ciclo;
- Larghezza dell'impulso: 7 s;
- Tempo di salita: 5 ns;
- Tensione di amplificazione: 4V

Solitamente per dei test più complessi vengono scritti dei test cases che si occupano di gestire in modo automatizzato e sincronizzato i vari strumenti da laboratorio pertanto non si hanno problemi di velocità nell'acquisizione di misure. In tal caso, invece, le misure vengono eseguite accendendo manualmente tutta la strumentazione e per tale motivo è necessario impostare un periodo di 10 secondi ed una larghezza dell'impulso di 7 secondi.
Per misurare la corrente con l'oscilloscopio, si è utilizzata una **sonda di corrente**: le sonde di corrente sono progettate per offrire un modo comodo di misurare la corrente e possono essere accoppiate in c.a., utilizzando la tecnologia dei trasformatori di corrente, oppure possono essere accoppiate in c.a./c.c. In entrambi i casi hanno una geometria a nucleo apribile che facilita la chiusura del conduttore elettroportante senza doverlo dissaldare.

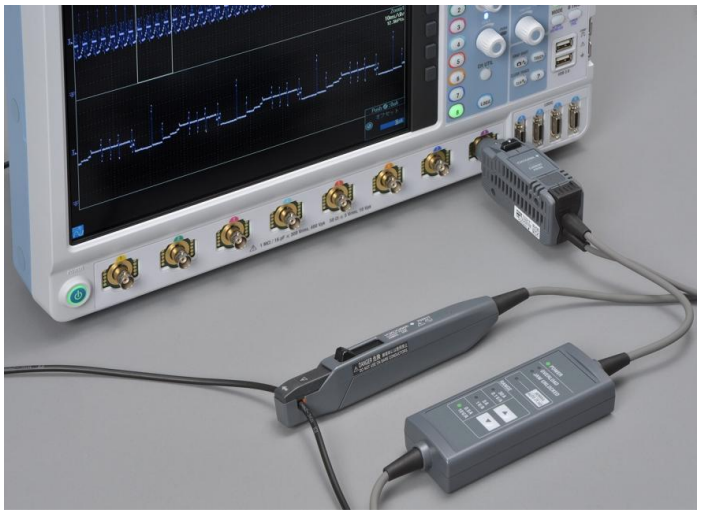


Figura 6.14: Sonda di corrente

Più precisamente, il source del mosfet è collegato alla sonda di corrente ed al ground di uscita della PCB (GND_out) e la sonda di corrente è poi connessa all'oscilloscopio per consentire di visualizzare le forme d'onda.
La corrente I_{OUT} misurata dalla sonda di corrente, come mostrato in fig. 6.12 in uscita dal source, è visualizzata sul canale 2 dell'oscilloscopio; sullo stesso canale viene anche impostato il trigger in modalità 'normal'.
In uscita all'LDO è connesso anche un carico pari a 115 Ω: quando il mosfet si accende, si ha che la corrente I_{OUT} scorre su un carico che risulta essere pari al parallelo delle due resistenze: $12.2\Omega//115\Omega \simeq 11\ \Omega$, ottenendo così un valore di corrente maggiore di 400 mA che, pertanto, segnalerà presenza di **overcurrent**.
In seguito, si ha la condizione in cui la corrente in uscita dall'LDO diminuisce poiché si verifica la riapertura del mosfet, che prima era chiuso per poter permettere il passaggio di corrente: quando accade ciò, la corrente I_{OUT} torna a scorrere sul carico di 115 Ω riportando il segnale di overcurrent basso.
Dopo aver impostato il banco di prova per il test di OVC, si procede, come per il segnale di UV spiegato nel sottocapitolo precedente, con il collegamento del pin di OVC della board con il corrispettivo pin DIO11 dell'FPGA (oltre al collegamento del pin di enable connesso al pin DIO12 dell'FPGA).

Dopo aver opportunamente collegato i pin della PCB con i pin dell'FPGA ed aver impostato in modo corretto tutti i vari strumenti di misura, si procede abilitando la batteria virtuale premendo il bottone 'START_BAT' rappresentato in fig. 6.7 ed abilitando l'LDO tramite l'interfaccia per il microcontrollore di Infineon (fig. 6.8), cliccando sul codice 'init_and_enable' → 'Execute'.
In seguito è riportata l'immagine ottenuta dall'oscilloscopio che evidenzia tutti i segnali relativi all'esecuzione del test di OVC.

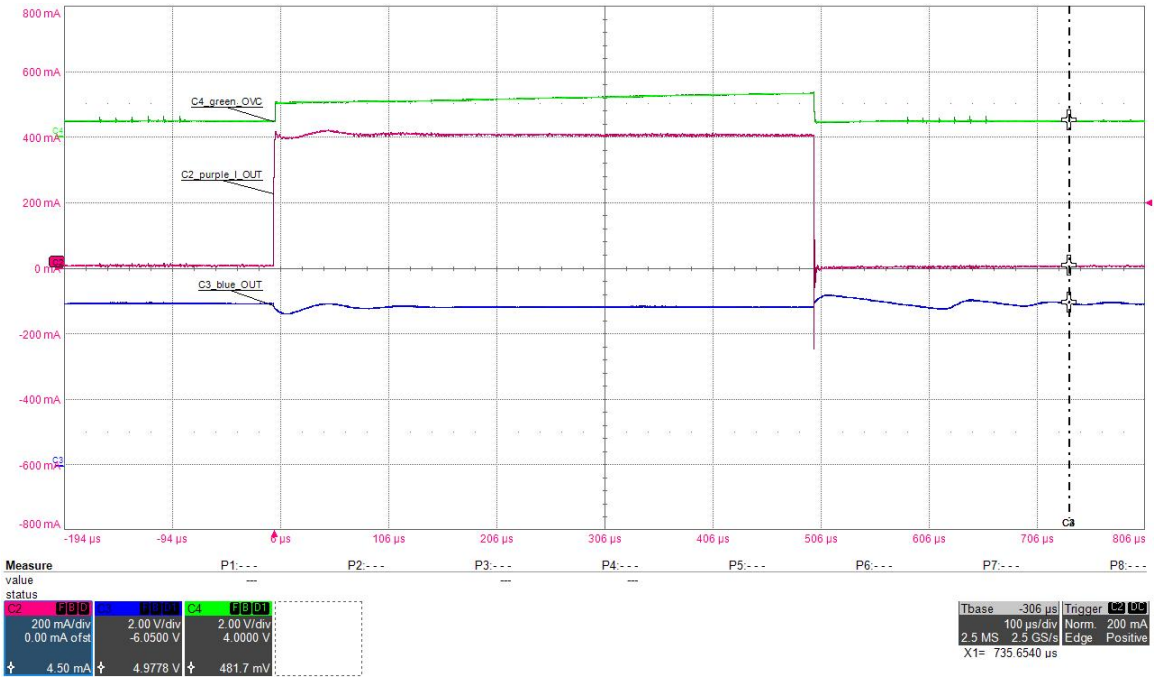


Figura 6.15: Controllo di OVC

In particolare:

- Sul canale 2 → la corrente $\mathbf{I_{OUT}}$ in uscita dal mosfet;
- Sul canale 3 → l'**uscita** dell'**LDO**;
- Sul canale 4 → il segnale di **OVC**.

Non appena la corrente I_{OUT} supera i 400 mA, il segnale di OVC diventa alto portando la tensione in uscita dall'LDO ad un valore minore di 5 V.

Il valore della tensione in uscita dall'LDO si riassesta non appena la corrente I_{OUT} diminuisce, riportando, quindi, il segnale di OVC basso.
L'ultimo passaggio prevede il controllo dei registri dell'FPGA: in particolare, dallo studio della parte digitale effettuata dall'altro tesista, è emerso che dovrebbe cambiare il bit in posizione 0 del registro 0x69.
Dallo schema ottenuto dall'interfaccia per il microcontrollore di Infineon risulta, infatti, che il bit in posizione 0 del registro 0x69 è alto segnalando, pertanto, presenza di OVC.

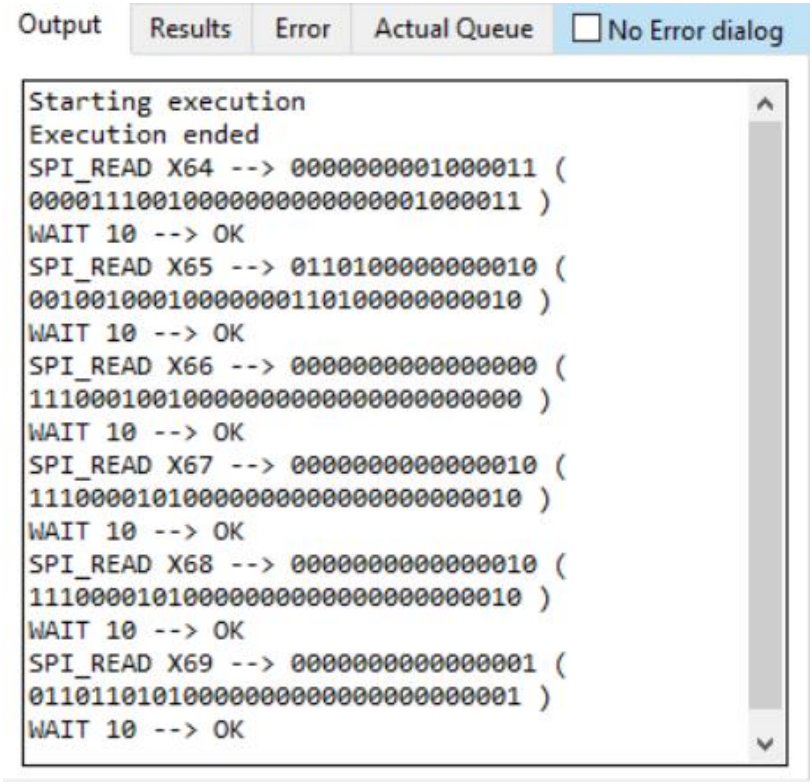


Figura 6.16: Lettura registri dell'FPGA per segnale di OVC

Il loop di verifica **PCB-FPGA** risulta quindi concluso e funzionante anche nel caso del segnale di overcurrent.

6.6 Test per il controllo di overtemperature (TS&TW)

Per effettuare test relativi al controllo di temperatura è stato necessario utilizzare il thermostream T-2500E (descritto nel capitolo 3.4).
Per prima cosa vengono collegate le sonde dell'oscilloscopio in tal modo:

- Sul canale 1 → segnale di **TW**;
- Sul canale 3 → **uscita** dell'**LDO**;
- Sul canale 4 → segnale di **TS**.

Si ricordi, inoltre, che la **temperatura di apertura** dei due interruttori meccanici utilizzati per il controllo di overtemperature è pari a **70**°.
Pertanto, per una migliore visualizzazione, si effettuano due tipologie di test (sia per il caso del TW sia per il caso del TS):

a) Il primo test effettuato per il TW prevede l'impostazione del trigger sul canale 1 in modalità '**positive edge**' cioè con pendenza positiva, mentre il secondo test per il TW prevede l'impostazione del trigger sul canale 1 in modalità '**negative edge**' cioè con pendenza negativa;

b) Il primo test effettuato per il TS prevede l'impostazione del trigger sul canale 4 in modalità '**positive edge**' cioè con pendenza positiva, mentre il secondo test per il TS prevede l'impostazione del trigger sul canale 4 in modalità '**negative edge**' cioè con pendenza negativa.

Per la prima tipologia di test per i segnali di TW e TS con trigger in modalità '**positive edge**', si procede impostando una temperatura di **75**° sulla schermata di controllo manuale del thermostream.

Figura 6.17: Schermata di controllo manuale del thermostream

Poi si posiziona il cappuccio termico conduttivo del thermostream direttamente sulla parte della PCB che contiene l'LDO e i due termostati: ciò consente di raggiungere in modo preciso le temperature di prova.

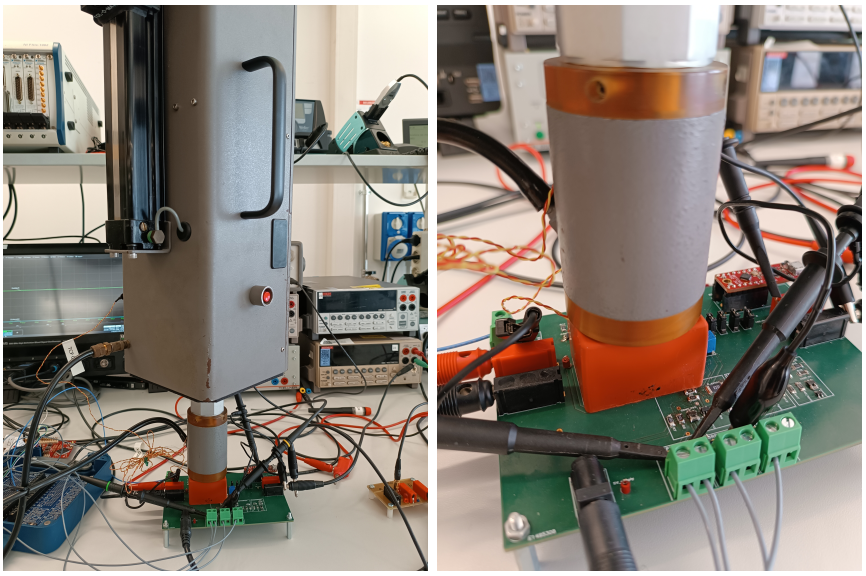


Figura 6.18: Posizionamento cappuccio termico sulla parte di PCB da testare

Come si è visto per i segnali di UV e OVC, anche in tal caso si effettua il collegamento dei pin relativi al TW e TS con i pin DIO dell'FPGA (oltre al collegamento del pin di enable connesso al pin DIO12), in particolare:

- Pin di TW connesso al DIO9;
- Pin di TS connesso al DIO10.

In seguito si procede con l'abilitazione della batteria virtuale e del segnale di enable tramite l'interfaccia del microcontrollore di Infineon.
In tal modo sarà possibile visualizzare il cambiamento della forma d'onda relativa al segnale di TS o TW che passerà da uno stato basso ad uno stato alto, a dimostrazione del corretto funzionamento dell'interruttore meccanico che si apre quando raggiunge la temperatura di riferimento.

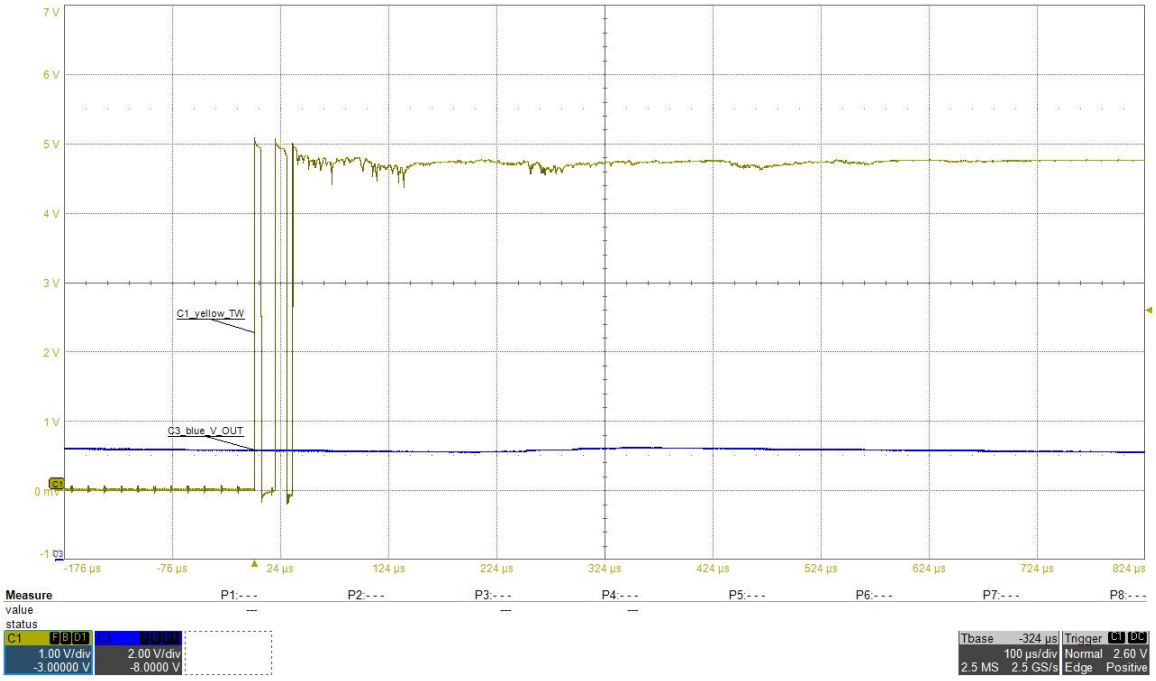


Figura 6.19: Controllo di TW con trigger in modalità 'positive edge'

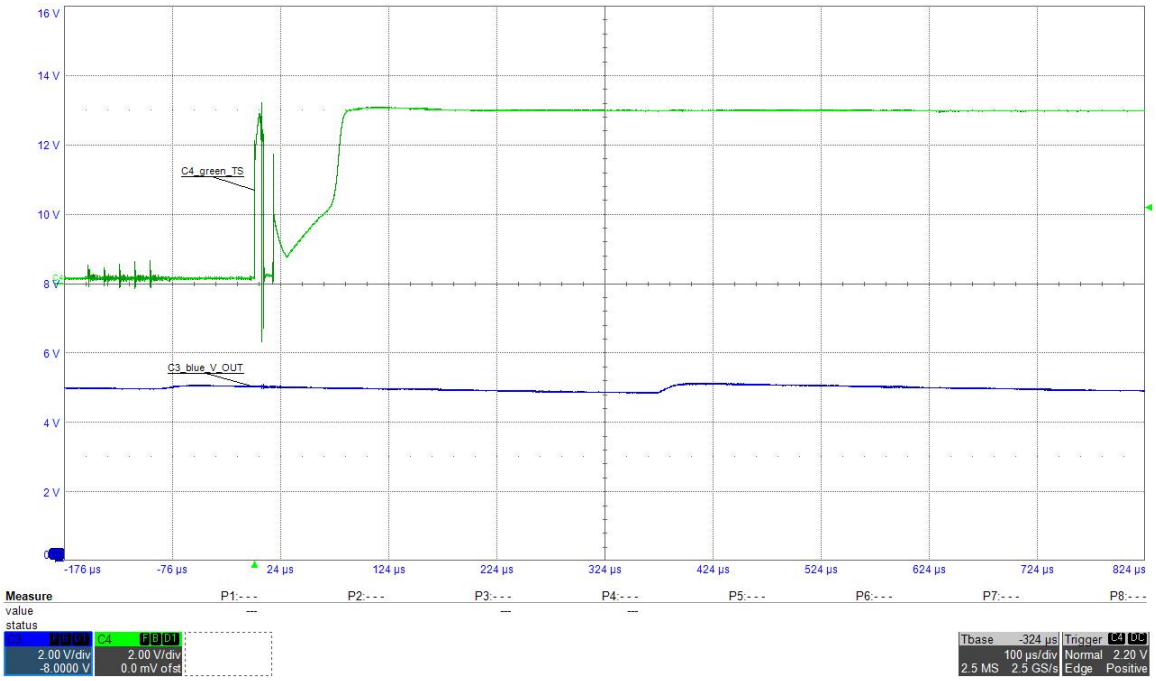


Figura 6.20: Controllo di TS con trigger in modalità 'positive edge'

L'ultimo step prevede la lettura ed il controllo dei registri dell'FPGA che fanno riferimento al segnale di TW e TS, nel caso di trigger positivo: in particolare, affinchè si rilevi un evento riguardante la variazione della temperatura di riferimento, deve verificarsi il cambiamento del bit in posizione 0 del registro 0x68 per il **TW** e del bit in posizione 1 del registro 0x69 per il **TS**.

Come si nota nelle figure sottostanti, l'evento di **TW** e di **TS** è segnalato dal cambiamento del bit corretto dei rispettivi registri.

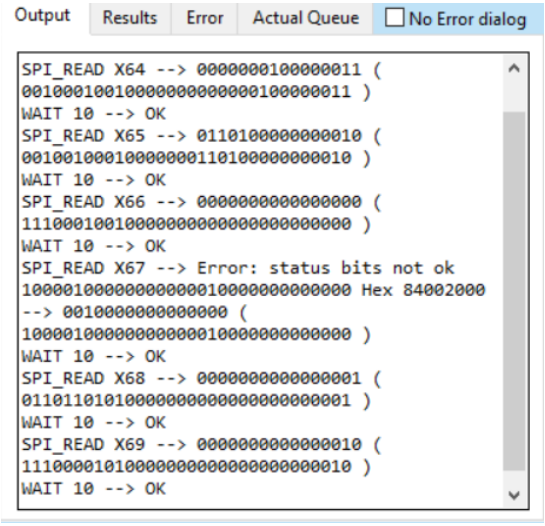


Figura 6.21: Lettura registri dell'FPGA per segnale di TW con trigger in modalità 'positive edge'

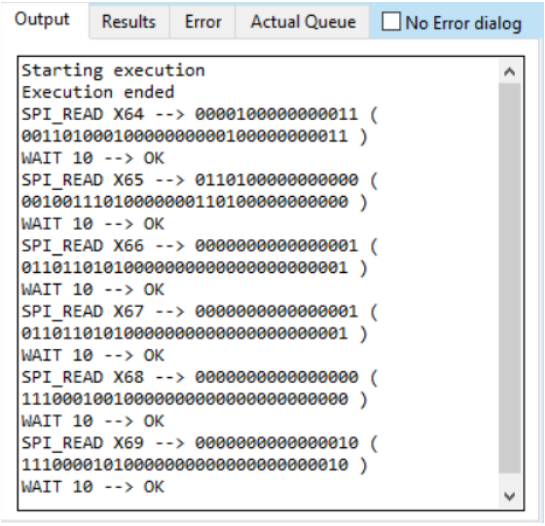


Figura 6.22: Lettura registri dell'FPGA per segnale di TS con trigger in modalità 'positive edge'

Anche per il controllo di overtemperature, pertanto, lo schema **PCB-FPGA** risulta funzionante.

Per la seconda tipologia di test per i segnali di TW e TS con trigger in modalità '**negative edge**', si procede considerando una temperatura ambiente, pari a **18**°, sulla schermata di controllo manuale del thermostream cliccando sul pulsante 'AMB' (come si nota in Fig. 6.17).

Tale test viene eseguito partendo dalla temperatura iniziale di 75° ed impostando il trigger negativo. Successivamente si setta la temperatura a 18°, valore della temperatura ambiente poiché in tal modo sarà possibile osservare il gradino in discesa sull'oscilloscopio per dimostrare la chiusura dell'interruttore meccanico non appena la temperatura scende sotto il valore di riferimento (70°).

Si posiziona il cappuccio termico conduttivo del thermostream direttamente sulla parte della PCB che contiene l'LDO e i due termostati, come mostrato in Fig. 6.18 e si procede abilitando la batteria virtuale su LabVIEW ed il segnale di enable tramite l'interfaccia del microcontrollore.

Di seguito sono riportati i risultati ottenuti: il segnale di TW o di TS passa da uno stato alto ad uno stato basso a causa del cambiamento di temperatura che varia da 75° a 18°, indicando la chiusura dell'interruttore quando la temperatura scende al di sotto della soglia.

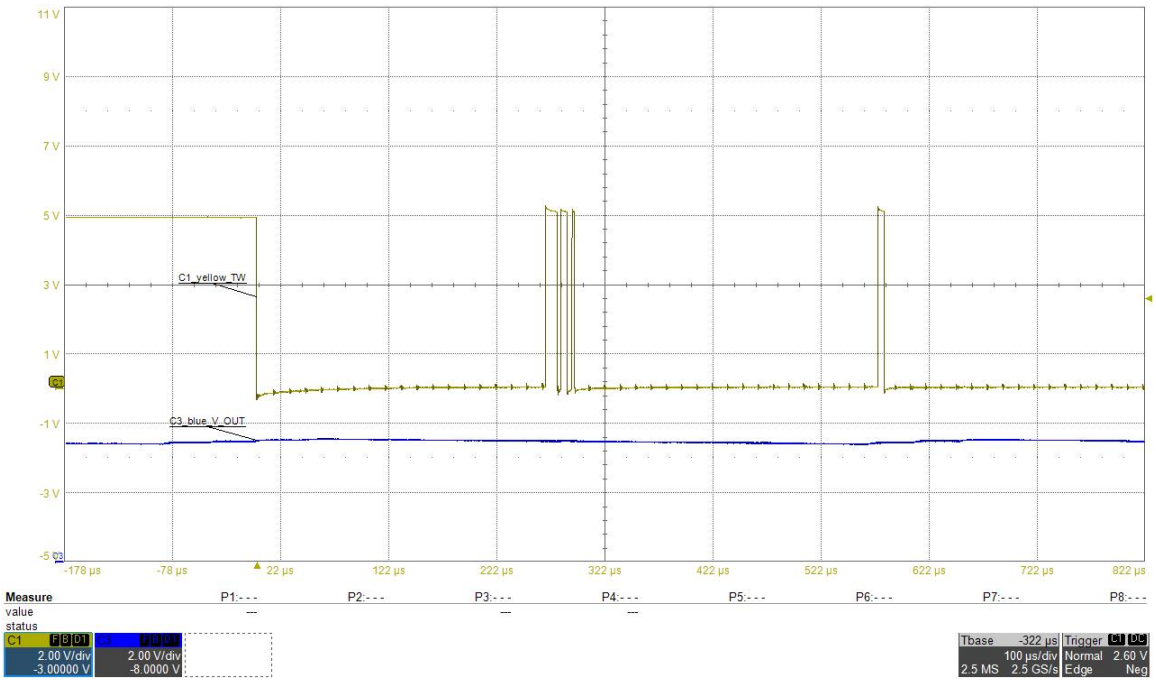


Figura 6.23: Controllo di TW con trigger in modalità 'negative edge'

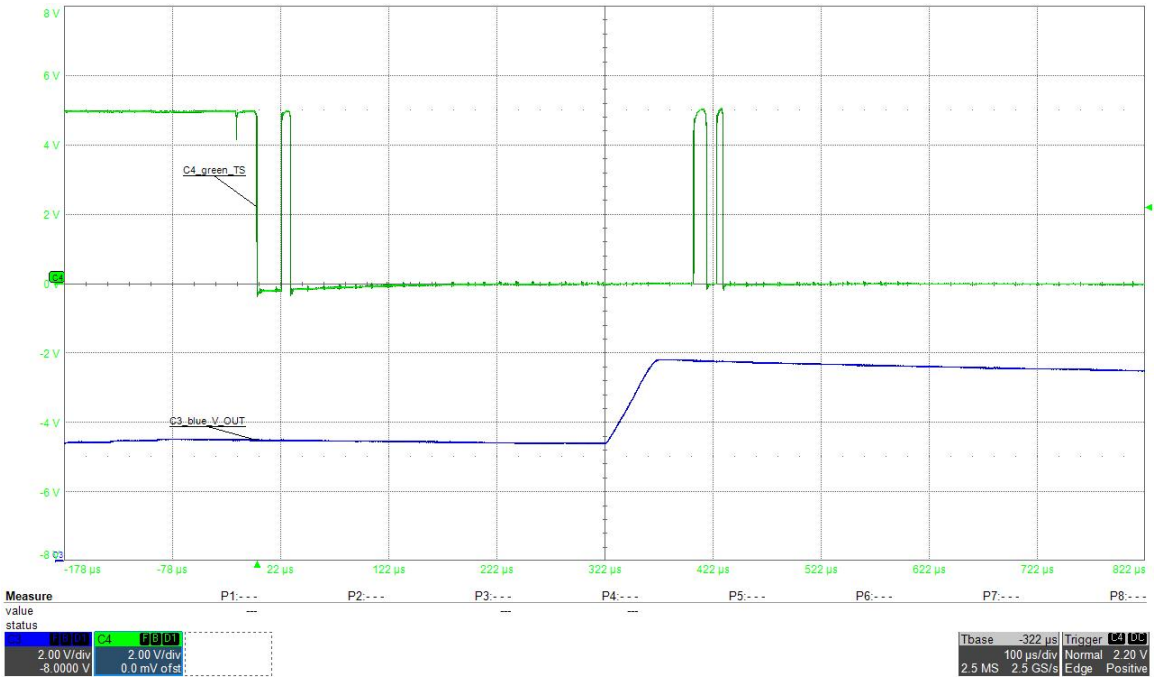


Figura 6.24: Controllo di TS con trigger in modalità 'negative edge'

Le oscillazioni presenti sui segnali di TW e TS sono dovute al fatto che i termostati utilizzati sono degli interruttore meccanici, pertanto potrebbero esserci delle piccole scariche che portano alla ri-apertura del dispositivo. Tuttavia, tali oscillazioni sono accettabili in quanto sono dell'ordine di qualche microsecondo.

Come per i test precedenti con trigger positivo, l'ultimo step prevede la lettura ed il controllo dei registri dell'FPGA che fanno riferimento al segnale di TW e TS considerando il trigger negativo; tuttavia l'altro tesista non ha preso in considerazione tale caso, pertanto non è stato possibile valutare la bontà dello schema PCB-FPGA per tale tipologia di test.

6.7 Test per il controllo di OV

I sistemi ed i dispositivi elettrici, durante la fase di progettazione, vengono dimensionati per funzionare ad una certa tensione elettrica, chiamata generalmente tensione nominale o tensione d'esercizio.
Nel caso in cui la tensione reale risulti essere superiore a quella nominale (che in tal caso risulta essere pari a 5 V) si verifica la condizione di overvoltage.
Per valutare tale condizione consideriamo i seguenti collegamenti con l'oscilloscopio:

- Sul canale 3 → **uscita** dell'**LDO**;
- Sul canale 4 → segnale di **OV**.

In seguito, si considera sul canale 4 il trigger in modalità 'normal'.
Per simulare una condizione di overvoltage, viene collegato il sourcemeter all'uscita dell'LDO per fornirgli una tensione maggiore della tensione di funzionamento. Nella figura sottostante si può notare, infatti, una tensione fornita pari a 5.5 V:

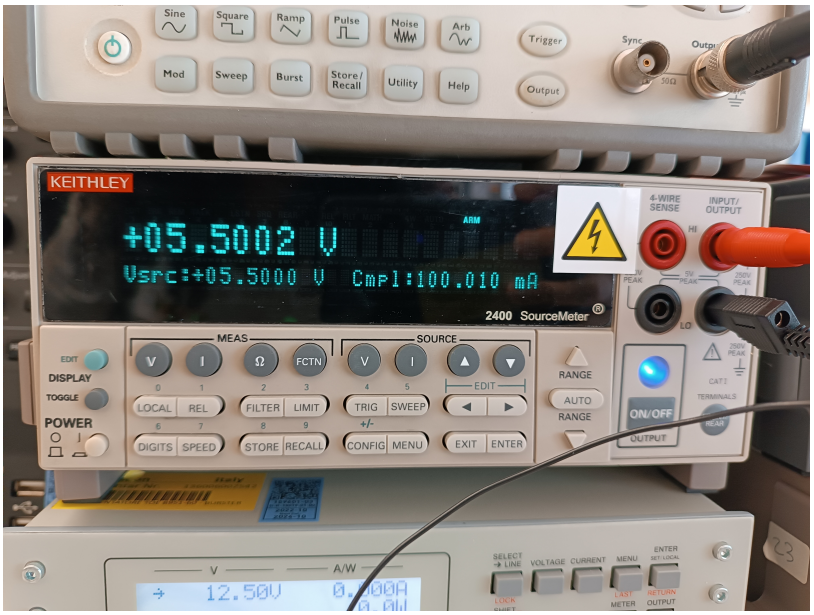


Figura 6.25: Valore di tensione maggiore della tensione di funzionamento dell'LDO

Dalla Fig. 6.26 si nota che, non appena LDO supera la tensione di un valore pari a circa 5.3 V, il segnale di overvoltage diventa alto dimostrando così il corretto funzionamento anche per tale tipologia di controllo.

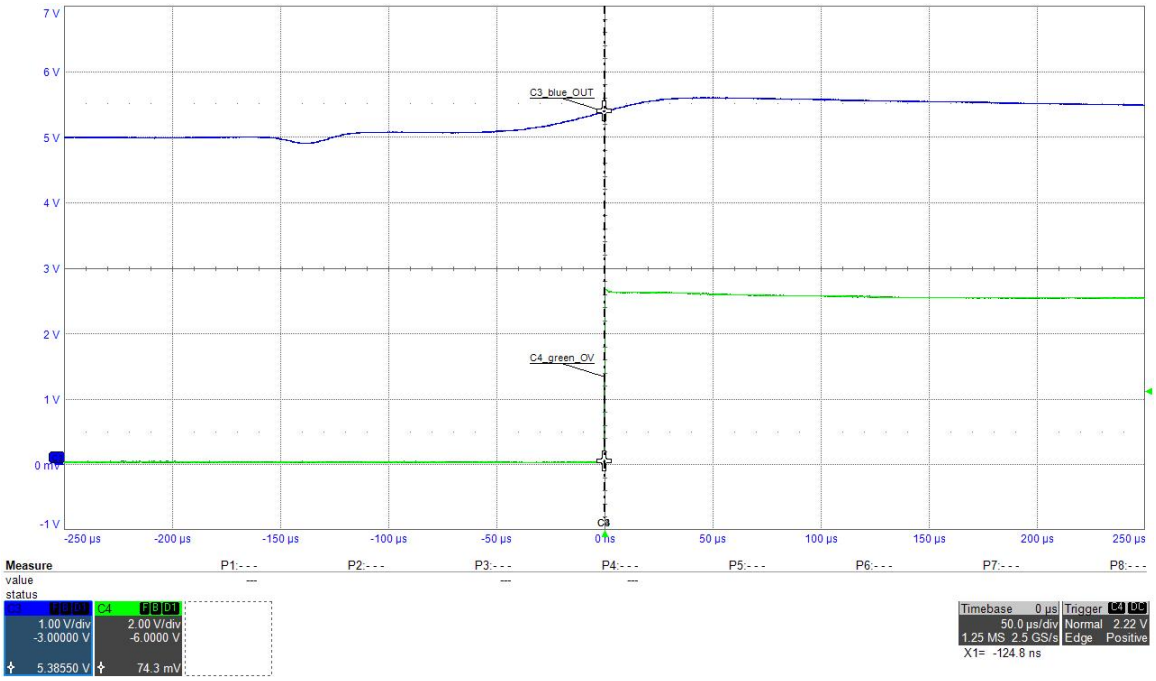


Figura 6.26: Controllo di OV

L'ultimo passaggio prevede la lettura ed il controllo dei registri dell'FPGA che fanno riferimento al segnale di OV. Tuttavia, il tesista che ha curato la parte digitale non ha considerato il segnale di OV pertanto non è stato possibile valutare il collegamento con l'FPGA per tale tipologia di test.

Capitolo 7

Conclusioni

Giunti alla fine del progetto si può affermare che la realizzazione della connessione di una scheda PCB con la relativa interfaccia digitale sintetizzata in una scheda FPGA si considera riuscita, come verificato dai test descritti nel capitolo precedente.
Per tale lavoro di tesi era prevista l'emulazione di un solo post-regolatore del PMIC pertanto è ovvio pensare che, se le specifiche dovessero cambiare o se si considerassero più LDO, lo sforzo per integrarli sarebbe minimo. Questo rende il sistema molto efficace come ambiente di prototipazione, in quanto i progetti possono essere iterati rapidamente.
In questa tesi è stato pertanto possibile realizzare una PCB che fosse in grado di fungere da interfaccia hardware, attraverso modelli di simulazione analogica, per la prototipazione di chip virtuali.
Implementando tutti i controlli esterni all'LDO è possibile, quindi, non solo interfacciarsi con la logica integrata nell'FPGA, che rappresenta la logica di controllo dell'LDO, ma anche iniziare a fare dei primi test di verifica senza dover necessariamente aspettare il PMIC finale: l'importanza di tale progetto, infatti, riguarda soprattutto il potenziale del suo utilizzo per l'accelerazione delle verifiche in laboratorio.
Secondo gli ingegneri di verifica dei componenti di Infineon, il tempo per la simulazione logica richiesto per verificare un chip richiede 10000 secondi, ovvero circa 2.5 ore. Tale interfaccia di prototipazione, insieme alla parte digitale dell'FPGA, migliora enormemente i tempi di verifica di un progetto e consente di procedere molto più

rapidamente.

A livello digitale, la porta di uscita potrebbe essere ottimizzata per inviare, al momento dell'aggiornamento, solo singoli e specifici registri invece di scaricare l'intero banco.

In questa tesi è stato principalmente dimostrato che il sistema è in grado di funzionare con la scheda analogica fisica realizzata. Il sistema non era tuttavia privo di problemi, il che non è del tutto inaspettato dal punto di vista dello sviluppo.

In conclusione il progetto è riuscito a fornire una prova pienamente funzionante per una piattaforma virtuale: questi risultati, pertanto, incoraggiano ad integrare la metodologia di piattaforma virtuale nel flusso di progettazione e sviluppo dei prodotti Infineon.

Capitolo 8

Sviluppi futuri

Il lavoro sviluppato in questi mesi ha già dimostrato la sua utilità e validità, tuttavia ogni progetto ha sempre un margine di miglioramento.
Solitamente i test eseguiti in laboratorio per grandi applicazioni prevedono la scrittura di varie righe di codice scritte in Matlab, denominate 'test cases', che permettono l'esecuzione del test e l'accensione degli strumenti di misura in maniera autonoma ed automatica: pertanto questa è sicuramente la prima sfida sulla quale poter lavorare.
Un altro miglioramento potrebbe riguardare, invece, i numerosi collegamenti che ci sono tra la PCB e l'FPGA, come ad esempio è mostrato in Fig. 6.4, che sono necessari per la connessione tra parte analogica e parte digitale in quanto, sebbene in questo progetto di tesi siano stati considerati solo cinque segnali di controllo per l'LDO, in futuro potrebbero essere richiesti ulteriori segnali di controllo e di conseguenza ci sarebbero altre connessioni.
Pertanto per aumentare l'efficienza del sistema, si potrebbe pensare di condensare i bus di comunicazione che collegano l'LDO con il core del PMIC, il quale è legato al core dell'FPGA.

Bibliografia

[1] Y. Ni, W. S. Mong, and J. Zhu, "On virtual prototyping of embedded system-on-chips," in *2011 9th IEEE International Conference on ASIC*, 2011, pp. 1106–1109, doi: 10.1109/ASICON.2011.6157402.

[2] R. Bhattacharya, S. Biswas, and S. Mukhopadhyay, "Fpga based chip emulation system for test development of analog and mixed signal circuits: A case study of dc-dc buck converter," in *Measurement*, 2012, pp. 1997–2020, doi: 10.1109/ASICON.2011.6157402.

[3] B. Deepaksubramanyan, P. Parakh, Z. Chen, H. Diab, D. Marcy, and F. Schlereth, "An fpga-based mos circuit simulator," in *48th Midwest Symposium on Circuits and Systems, 2005.*, 2005, pp. 655–658 Vol. 1, doi: 10.1109/MWSCAS.2005.1594186.

[4] What is LabVIEW? https://www.ni.com/it-it/shop/labview.html.

[5] PXIe-1082-NI. https://www.ni.com/it-it/support/model.pxie-1082.html.

[6] Il CLB delle FPGA Xilinx Virtex-5. https://it.emcelettronica.com/il-clb-delle-fpga-xilinx-virtex-5.

[7] Altium Designer Overview - PCB Design Tool. https://www.altium.com/it/altium-designer.

[8] Datasheet Toellner TOE8951. https://www.toellner.de/datenblaetter/en_8951_52.pdf.

[9] What's the difference between an SMU and a DMM with a power supply? https://www.electronicdesign.com/technologies/test-measurement/article/21795921/whats-the-difference-between-an-smu-and-a-dmm-with-a-power-supply.

[10] Datasheet Agilent 33250A. https://www.farnell.com/datasheets/1599391.pdf.

[11] Datasheet Thermostream T-2500E. https://www.atecorp.com/atecorp/media/pdfs/data-sheets/thermonics-t-2500se_datasheet.pdf.

[12] LabVIEW FPGA Compile System Overview. https://www.ni.com/it-it/shop/electronic-test-instrumentation/add-ons-for-electronic-test-and-instrumentation/what-is-labview-fpga-module/ni-labview-fpga-compilation-options.html.

[13] Datasheet LDO. https://www.infineon.com/dgdl/Infineon-TLE42764-DS-v01_30-EN.pdf?fileId=5546d46258fc0bc101595f8e8c751fa3.

[14] Comparatore con isteresi. https://www.demiacos.com/blog/comparatori-con-isteresi.

[15] Datasheet comparatore di tensione LM397MF/NOPB. https://www.analog.com/media/en/technical-documentation/data-sheets/ADM4073.pdf.

[16] Datasheet amplificatore di rilevamento corrente ADM4073TWRJZ-REEL7. https://www.analog.com/media/en/technical-documentation/data-sheets/ADM4073.pdf.

[17] Datasheet termostato a disco bimetallico subminiaturizzato. https://docs.rs-online.com/b126/0900766b813d95f8.pdf.

[18] XMC2GO and DAVE. https://it.emcelettronica.com/xmc2go-dave-how.

[19] XMC 2Go evaluation kit user guide. https://www.infineon.com/dgdl/Board_Users_Manual_XMC_2Go_Kit_with_XMC1100_R1.0.pdf?fileId=db3a3043444ee5dc014453d6c75078c6.