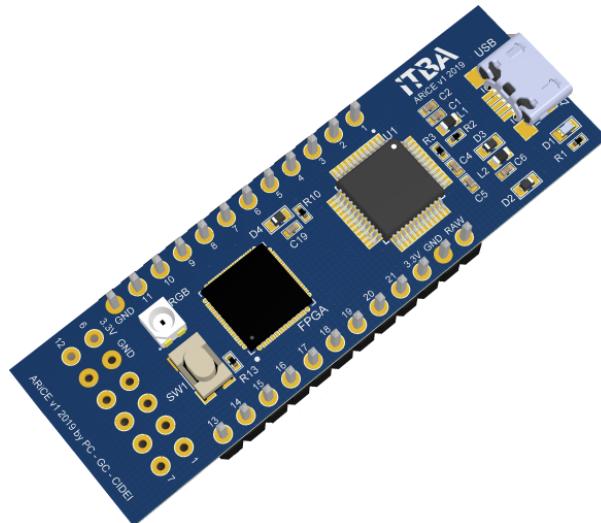


CENTRO DE INVESTIGACIÓN Y DESARROLLO EN
ELECTRÓNICA INDUSTRIAL (CIDEI)



Plataforma ARiCE

Primeros Pasos Apio



AUTORES: Dr. Ing. Pablo COSSUTTA - Ing. Gonzalo CASTELLI - Rodrigo DEVESA

CIUDAD AUTÓNOMA DE BUENOS AIRES
2018-2019

Contenido

1. Introducción	3
2. Primeros pasos	3
2.1. Descargando el software	3
2.2. Creando un nuevo proyecto	4
2.3. Programando la placa	5
2.4. Simulando el diseño	5
3. Información de las conexiones	6
4. Circuito Impreso Libre	8
4.1. Listado de componentes	9
4.2. Diagramas esquemáticos	9
5. Licencia	13
6. Agradecimientos	13

1. Introducción

Este proyecto es una plataforma de código abierto, la cual se muestra en la Fig. 1, basado en una placa FPGA de bajo costo y bajo consumo de [Lattice Semiconductor](#). Tiene como objetivo ser utilizada en una amplia gama de aplicaciones de procesamiento de señales y control, tanto para la educación como para la industria.

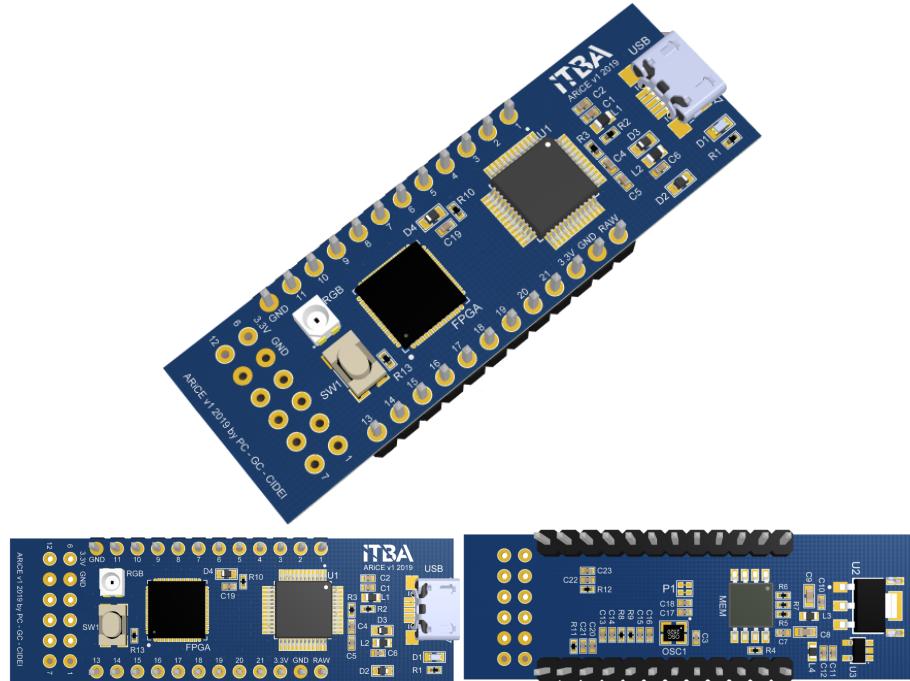


FIGURA 1: Vistas de la placa

2. Primeros pasos

Para comenzar a utilizar la plataforma hay que cumplir tres pasos. El primero es descargar e instalar el software, después crear un proyecto incluyendo los archivos de código necesarios y por último cargar el programa en la memoria interna del FPGA usando un cable USB.

2.1. Descargando el software

Antes de usar la placa, es necesario descargar el software para programar la FPGA Lattice iCE40-UP5K. En este tutorial se va a utilizar la distribución basada en [Apio Open Source Ecosystem for FPGAs](#), la cual permite sintetizar, implementar y verificar el código y también programar la FPGA mediante un puerto USB. [Python 3.7+](#) debe ser instalado previamente para poder instalar Apio.

Durante la instalación de Python debe asegurarse de seleccionar "**Customize Installation**". Luego, dentro de las opciones adicionales, seleccionar la instalación de *pip*, como se muestra en la Fig. 2, dado que se necesita para descargar e instalar el paquete de Apio.

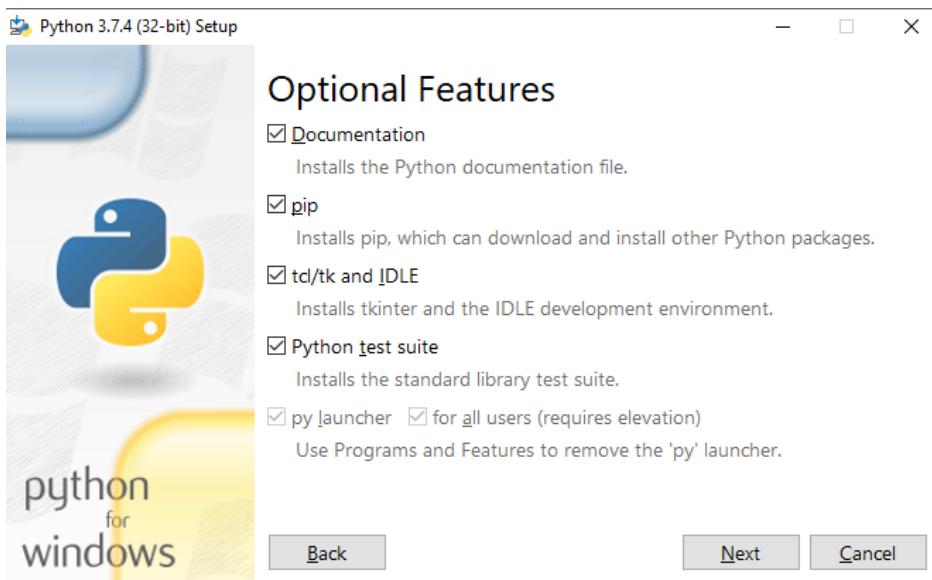


FIGURA 2: Configuración de la instalación de Python

Una vez finalizada la instalación de Python con pip, se procede a instalar Apio. En Windows, se debe abrir una consola de comandos e ingresar:

```
pip install -U apio
```

Cuando finaliza, es necesario instalar los paquetes adicionales de Apio utilizando el siguiente comando:

```
apio install --all
```

Este proceso puede demorar varios minutos.

2.2. Creando un nuevo proyecto

Una vez instalado el software, el siguiente paso es crear una nueva carpeta para el proyecto, y empezar a escribir el código. Un ejemplo de proyecto que utiliza el LED RGB montado en la placa está disponible en nuestro repositorio Github. Después de descargar los archivos, se colocan en una carpeta fácilmente accesible desde donde ejecutaremos el código.

Use el comando cd seguido por la dirección de la carpeta de proyecto para seleccionarla y trabajar sobre ella. Por ejemplo, si el proyecto está en una carpeta llamada "fpga_project." en el Escritorio, use:

```
cd Desktop/fpga_project
```

Una vez seleccionada la carpeta a utilizar, generamos el archivo de configuración .init adecuado para nuestra placa, ya que Apio soporta varias plataformas de FPGA diferentes. Para hacer esto, use el comando:

```
apio init --board upduino2
```

Ahora puede usar el comando:

```
apio build
```

para asegurar que el proyecto esta correctamente configurado. La carpeta debería contener al menos 3 archivos; un .v con el código del programa, un .pcf con información de entradas y salidas para el programa, y el archivo .init que acabamos de crear. Si el programa compila correctamente, puede pasar a programar la placa FPGA.

2.3. Programando la placa

Ahora que el proyecto esta armado, podemos cargar el programa a nuestra placa para ejecutarlo.

Conecte la FPGA a la PC con un cable USB, espere unos segundos para que Windows configure los controladores, y luego utilice el siguiente comando para programar la placa:

```
apio upload
```

El proceso puede demorar algunos segundos. Si todo funcionó correctamente, la placa debería configurarse desde la memoria flash interna y el LED comenzará a prenderse y a cambiar de color.

2.4. Simulando el diseño

Para simular el diseño ejecutar el siguiente comando:

```
apio sim
```

Se cargará el visualizador [GTKWave](#) y se podrá analizar el resultado de la simulación.

3. Información de las conexiones

La placa posee múltiples pines de E/S, junto con alimentaciones, distribuidas en los conectores laterales y frontales del circuito impreso. La siguiente lista provee una breve descripción de estas señales:

- IOT_XX corresponde a pines estándar de E/S. En el modo usuario, después de la configuración, estos pines pueden ser utilizados como E/S según el usuario y corresponden al banco superior (top, donde xx = ubicación de E/S)
- IOB_XX corresponde a pines estándar de E/S. En el modo usuario, después de la configuración, estos pines pueden ser utilizados como E/S según el usuario y corresponden al banco inferior (bottom, donde xx = ubicación de E/S)
- RAW VCC. La alimentación de la placa cuando utiliza una fuente de alimentación externa. La alimentación puede provenir tanto del conector USB como del pin RAW VCC (Tensión mínima de 4.5V y máxima de 6V)
- 3.3V. Salida de 3.3V generada por el regulador de tensión lineal interno. No excede los 500mA
- GND. Pines de referencia
- Las señales que contienen los sufijos G1, G3 y G6 pueden ser utilizadas tanto como pines estándar de E/S o señales globales con gran cantidad de conexiones internas o como líneas de clock o reset. Estos pines se encuentran conectados a los buffers globales GBUF1, GBUF3 y GBUF6 respectivamente.

La Tabla 1 muestra la correspondencia entre la asignación de pines, el nombre de las señales y el pad correspondiente al encapsulado de la FPGA.

TABLA 1: Conexiones

Pin Placa	Pin FPGA	Señal	Pin Placa	Pin FPGA	Señal
Conexiones Laterales			Conexiones Frontales		
1	20	IOB 25B G3	1	4	IOB 8A
2	21	IOB 23B	2	3	IOB 9B
3	23	IOT 37A	3	47	IOB 2A
4	25	IOT 36B	4	44	IOB 3B G6
5	26	IOT 39A	5	-	GND
6	27	IOT 38B	6	-	3.3V
7	31	IOT 42B	7	48	IOB 4A
8	32	IOT 43A	8	45	IOB 5B
9	34	IOT 44B	9	38	IOT 50B
10	36	IOT 48B	10	42	IOT 51A
11	37	IOT 45A G1	11	-	GND
12	-	GND	12	-	3.3V
13	2	IOB 6A	Pin Placa	Pin FPGA	Señal
14	6	IOB 13B	No Conectados		
15	9	IOB 16A	-	43	IOT 49A
16	10	IOB 18A	-	46	IOB 0A
17	11	IOB 20A	-	28	IOT 41A
18	12	IOB 22A	Color LED		
19	13	IOB 24A	Pin FPGA		
20	18	IOB 31B	Azul	39	RGB0
21	19	IOB 29B	Verde	40	RGB1
22	-	3.3V	Rojo	41	RGB2
23	-	GND			
24	-	RAW VCC			

Los pares diferenciales se muestran en la Tabla 2. Los mismos están agrupados y en colores, donde fondo blanco corresponde al terminal positivo y celeste al negativo.

TABLA 2: Pares Diferenciales

Pin Placa	Pin FPGA	Señal
3	23	IOT 37A
4	25	IOT 36B
5	26	IOT 39A
6	27	IOT 38B
8	32	IOT 43A
7	31	IOT 42B
11	37	IOT 45A G1
9	34	IOT 44B
2	21	IOB 23B
18	12	IOB 22A
2	3	IOB 9B
1	4	IOB 8A
4	44	IOB 3B G6
3	47	IOB 2A
8	45	IOB 5B
7	48	IOB 4A
10	42	IOT 51A
9	38	IOT 50B

4. Circuito Impreso Libre

los archivos de diseño, diagramas esquemáticos y el listado de componentes se encuentran disponibles en el repositorio de GitHub. Los archivos del proyecto realizado en Altium y los Gerbers de fabricación están disponibles. Además el diseño está publicado utilizando herramientas libres como [Circuit Maker](#) y [KiCad](#).

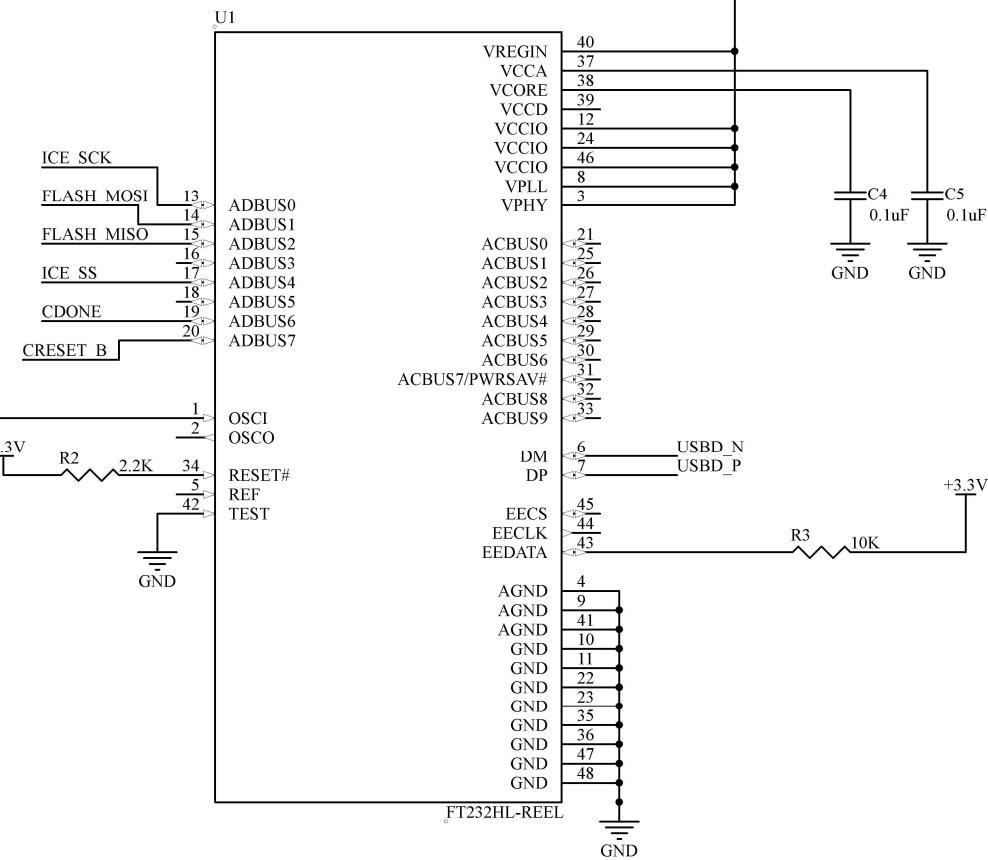
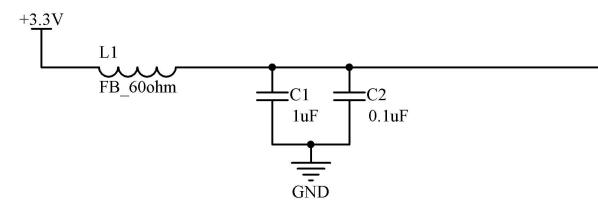
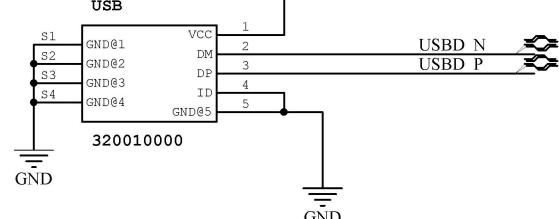
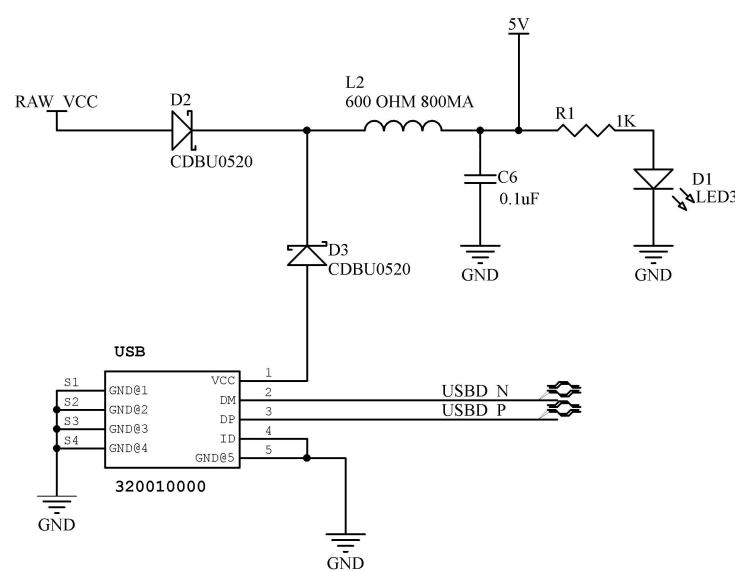
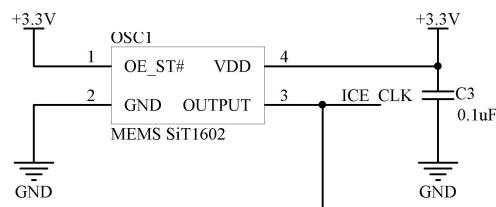
4.1. Listado de componentes

En la Tabla 3 se incluye todos los componentes de la plataforma ARiCE. Para mayor información se incluye el link con la mayoría de las hojas de datos de los componentes utilizados.

TABLA 3: Listado de componentes

Nombre	Cant.	Código del Fabricante	Valor	Footprint
C2, C3, C4, C5, C6, C10, C12, C14, C16, C18, C19, C21, C23	13	CL05B104KA5NNNC	0.1uF	0402
C8, C9	2	CC0603KRX5R8BB105	1uF	0603
C1, C7, C11, C13, C15, C17, C20, C22	8	CGB2A1JB1E105M033BC	1uF	0402
R1	1	RC0402JR-071KL	1KΩ	0402
R2	1x	RC0402FR-072K2L	2.2kΩ	0402
R3, R4, R5, R6, R7, R13	6	RC0402JR-0710KP	10kΩ	0402
R8, R9, R10, R11, R12	5	AC0402JR-071RL	1Ω	0402
L1	1	HI0603P600R-10	10mH	0603
L2, L3, L4	3	BLM18HE601SN1D	10mH	0603
RGB	1	CLMVC-FKA-CL1D1L71BB7C3C3	4-PLCC	
D1	1	LTST-C190TBKT		0603
U1	1	FT232HL-REEL	48-LQFP (7x7)	
U2	1	TLV1117LV33DCYR		SOT-223-4
U3	1	LP5907MFX-1.2/NOPB		SOT-23-5
OSC1	1	SIT1602AC-73-33S-12.000000G		2.0X1-6MM
FPGA	1	ICE40UP5K-SG48ITR50	48-QFN- 7X7	
USB	1	10118193-0001LF	Micro B SMD	USB
MEM	1	W25Q32JVSSIQ		SOP8
D2, D3, D4	3	CDBU0520		0603/SOD- 523F
SW1	1	PTS810 SJM 250 SMTR LFS		SW4-SMD

4.2. Diagramas esquemáticos



Title		FTDI Module	ITBA - CIDEI - PC - GC	Pablo Cossutta Gonzalo Castelli
Size	Number			Revision 2.0
A4				
Date: File:	12/14/2018 C:\Users...\FTDI.SchDoc		Sheet of Drawn By:	

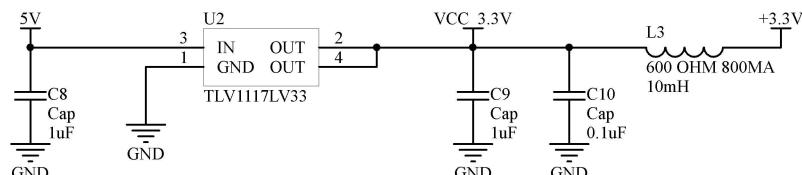
1

2

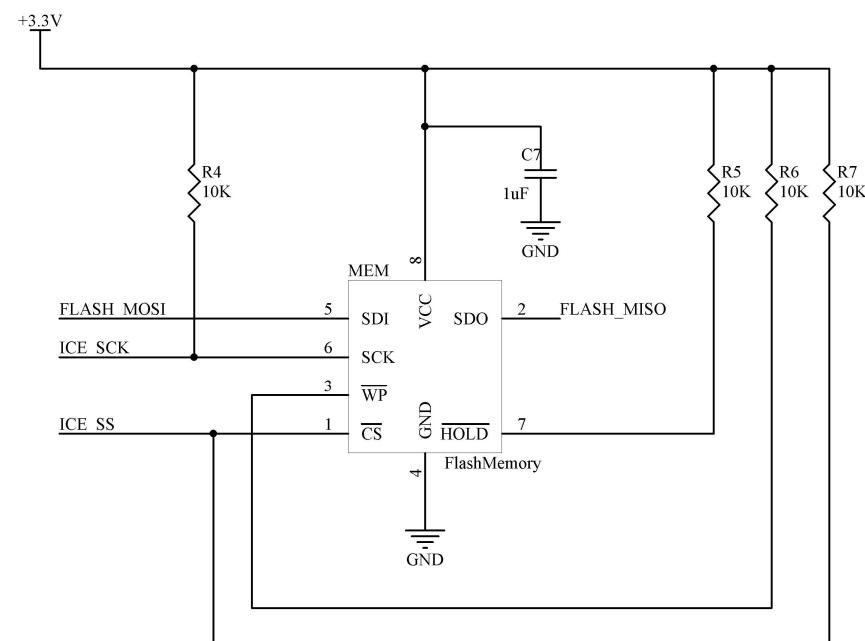
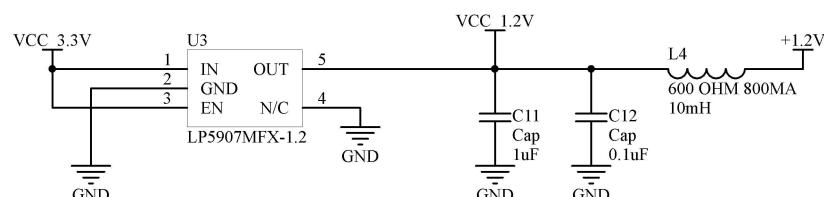
3

4

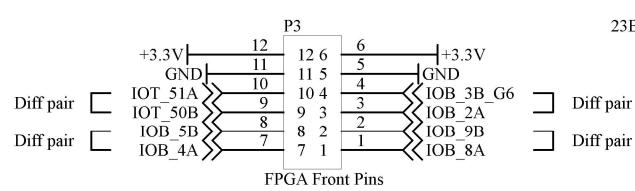
A



B

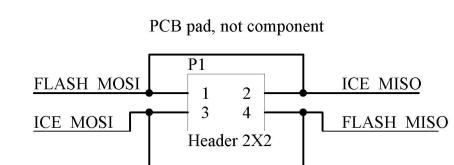
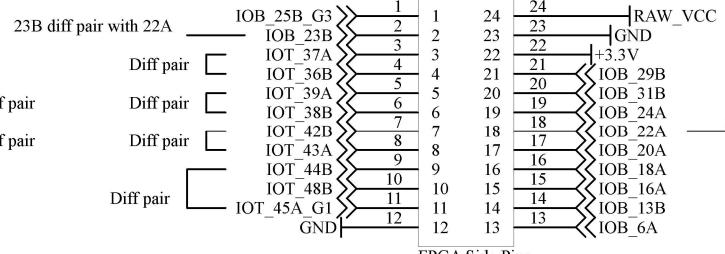


C



IOT_49A
IOB_0A
IOT_41A

Not Connected



To program FLASH (default, wired) =

To program iCE FPGA cut traces and solder ||

Title		FLASH Memory, Supply, Connectors ITBA - CIDEI - PC - GC	Pablo Cossutta Gonzalo Castelli
Size	Number		Revision 2.0
A4			
Date: 12/14/2018	Sheet of 1	File: C:\Users...\Misc.SchDoc	Drawn By:

1

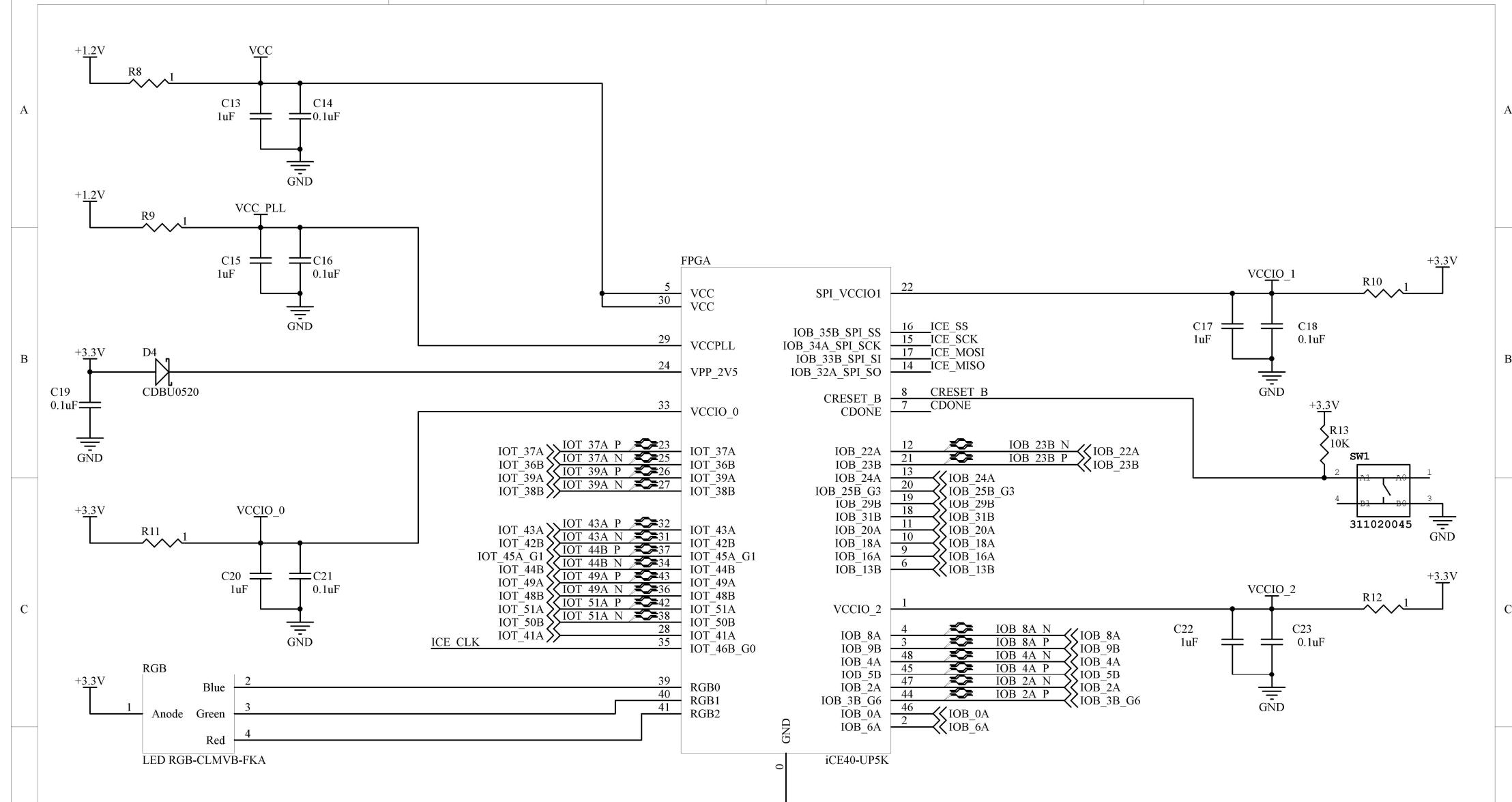
2

3

4

D

1 2 3 4



Title		FPGA	ITBA - CIDEI - PC - GC	Pablo Cossutta Gonzalo Castelli
Size		Number	Revision	
A4			2.0	
Date:		12/14/2018	Sheet of	
File:		C:\Users...\FPGA.SchDoc	Drawn By:	

1 2 3 4

5. Licencia

Este proyecto se provee bajo la licencia *Creative Commons Attribution Share-Alike*, lo que significa que puede ser utilizado libremente, adaptado a sus necesidades, sin necesidad de solicitar o pagar un permiso, siempre y cuando se denote apropiadamente los créditos del creador original y se libere el diseño bajo la misma licencia. Para mayor información visitar el sitio web [Creative Commons](#).

6. Agradecimientos

Este proyecto se inspira y basa en los manuales oficiales y las herramientas detalladas en [iCE40 UltraPlus Breakout Board](#) de Lattice Semiconductors y en la documentación de la FPGA, iCE40UP5K.