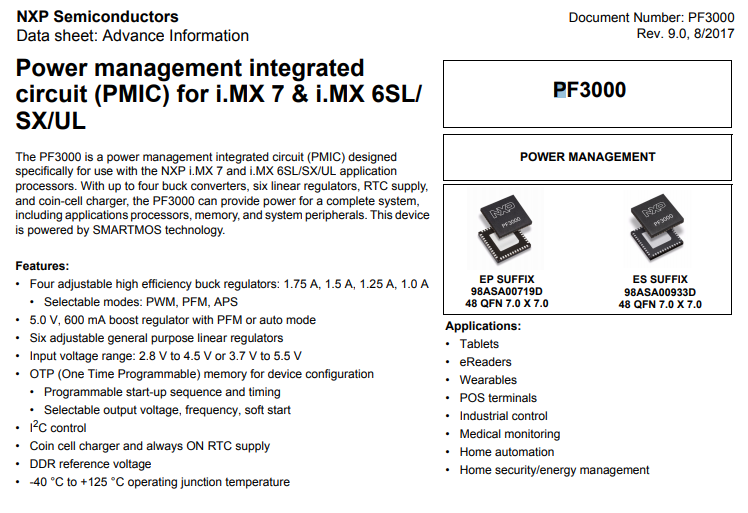
PMIC - Power management integrated circuit (PMIC) for i.MX 7 & i.MX 6SL/ SX/UL

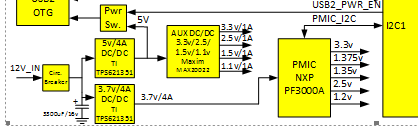
The board will use:

NXP PF3000A

<https://www.nxp.com/docs/en/data-sheet/PF3000.pdf>



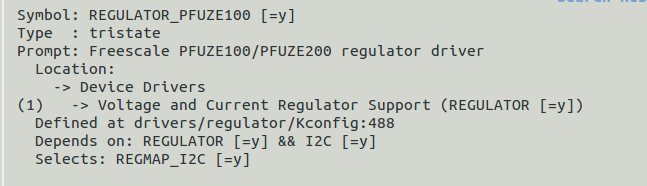
The PMIC is connected to I2C #1



And it supply 3.3, 1,375, 1.35, 2.5 and 1.2v.

**Kernel config**

The device driver is already enabled in the kernel



File:

bsp/linux/drivers/regulator/pfuze100-regulator.c

We can see again that that driver support several regulator ids:

static const struct i2c\_device\_id pfuze\_device\_id[] = {

{.name = "pfuze100", .driver\_data = PFUZE100},

{.name = "pfuze200", .driver\_data = PFUZE200},

{.name = "pfuze3000", .driver\_data = PFUZE3000},

{ }

And the compatible driver we need in the device tree is:

static const struct of\_device\_id pfuze\_dt\_ids[] = {

{ .compatible = "fsl,pfuze100", .data = (void \*)PFUZE100},

{ .compatible = "fsl,pfuze200", .data = (void \*)PFUZE200},

{ .compatible = "fsl,pfuze3000", .data = (void \*)PFUZE3000},

{ }

**Device tree**

The EVK currently is using

pmic: pf0100@08 {

compatible = "fsl,**pfuze100**";

We need the **pfuze3000**

Here we can see an example of a DTS with that chip.

<https://github.com/torvalds/linux/blob/master/arch/arm/boot/dts/imx7d-sdb.dts>

One question, if I will change it in the EVK just now, does the EVK will work?

Of course it should be reside in i2c 1

This is the new pmic configurations from the above dts link

|  |
| --- |
| &i2c1 { |
|  | pinctrl-names = "default"; |
|  | pinctrl-0 = <&pinctrl\_i2c1>; |
|  | status = "okay"; |
|  |  |
|  | pmic: pfuze3000@8 { |
|  | compatible = "fsl,pfuze3000"; |
|  | reg = <0x08>; |
|  |  |
|  | regulators { |
|  | sw1a\_reg: sw1a { |
|  | regulator-min-microvolt = <700000>; |
|  | regulator-max-microvolt = <1475000>; |
|  | regulator-boot-on; |
|  | regulator-always-on; |
|  | regulator-ramp-delay = <6250>; |
|  | }; |
|  |  |
|  | /\* use sw1c\_reg to align with pfuze100/pfuze200 \*/ |
|  | sw1c\_reg: sw1b { |
|  | regulator-min-microvolt = <700000>; |
|  | regulator-max-microvolt = <1475000>; |
|  | regulator-boot-on; |
|  | regulator-always-on; |
|  | regulator-ramp-delay = <6250>; |
|  | }; |
|  |  |
|  | sw2\_reg: sw2 { |
|  | regulator-min-microvolt = <1500000>; |
|  | regulator-max-microvolt = <1850000>; |
|  | regulator-boot-on; |
|  | regulator-always-on; |
|  | }; |
|  |  |
|  | sw3a\_reg: sw3 { |
|  | regulator-min-microvolt = <900000>; |
|  | regulator-max-microvolt = <1650000>; |
|  | regulator-boot-on; |
|  | regulator-always-on; |
|  | }; |
|  |  |
|  | swbst\_reg: swbst { |
|  | regulator-min-microvolt = <5000000>; |
|  | regulator-max-microvolt = <5150000>; |
|  | }; |
|  |  |
|  | snvs\_reg: vsnvs { |
|  | regulator-min-microvolt = <1000000>; |
|  | regulator-max-microvolt = <3000000>; |
|  | regulator-boot-on; |
|  | regulator-always-on; |
|  | }; |
|  |  |
|  | vref\_reg: vrefddr { |
|  | regulator-boot-on; |
|  | regulator-always-on; |
|  | }; |
|  |  |
|  | vgen1\_reg: vldo1 { |
|  | regulator-min-microvolt = <1800000>; |
|  | regulator-max-microvolt = <3300000>; |
|  | regulator-always-on; |
|  | }; |
|  |  |
|  | vgen2\_reg: vldo2 { |
|  | regulator-min-microvolt = <800000>; |
|  | regulator-max-microvolt = <1550000>; |
|  | }; |
|  |  |
|  | vgen3\_reg: vccsd { |
|  | regulator-min-microvolt = <2850000>; |
|  | regulator-max-microvolt = <3300000>; |
|  | regulator-always-on; |
|  | }; |
|  |  |
|  | vgen4\_reg: v33 { |
|  | regulator-min-microvolt = <2850000>; |
|  | regulator-max-microvolt = <3300000>; |
|  | regulator-always-on; |
|  | }; |
|  |  |
|  | vgen5\_reg: vldo3 { |
|  | regulator-min-microvolt = <1800000>; |
|  | regulator-max-microvolt = <3300000>; |
|  | regulator-always-on; |
|  | }; |
|  |  |
|  | vgen6\_reg: vldo4 { |
|  | regulator-min-microvolt = <2800000>; |
|  | regulator-max-microvolt = <2800000>; |
|  | regulator-always-on; |
|  | }; |
|  | }; |
|  | }; |
|  | }; |

From the table in

### Power supply voltages details

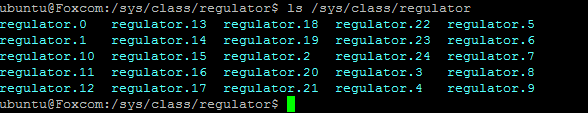
We definitely can see that the PMIC fits to the above DTS.

So we need to replace the current DTS with the above DTS for PMIC!

**Accessing from the Sysfs.**

Once the device boot up we can look using sysfs interface at

/sys/class/regulator



We can see that the regulator have name of regulator.X

We can look into and find the name:

cat /sys/class/regulator/regulator.11/name

SW1C

This example is from the EVK and not from the real board

From the device tree we can see:

reg\_vddsoc: sw1c { /\* VDDSOC\_IN \*/

regulator-min-microvolt = <300000>;

regulator-max-microvolt = <1875000>;

regulator-always-on;

};

Type:

cat /sys/class/regulator/regulator.11/type

voltage

cat /sys/class/regulator/regulator.11/max\_microvolts

1875000

Which fits

In regulator.12 we have always on and in regulator .16 we don’t

ls regulator.12/of\_node

name regulator-always-on regulator-max-microvolt regulator-min-microvolt

The complete sysfs interface can be found here

<https://www.kernel.org/doc/Documentation/ABI/testing/sysfs-class-regulator>

The EVK pmic regulator device does not have state rather then

what: /sys/class/regulator/.../suspend\_standby\_state

Date: May 2008

KernelVersion: 2.6.26

Contact: Liam Girdwood <lrg@slimlogic.co.uk>

Description:

Some regulator directories will contain a field called

suspend\_standby\_state. This holds the regulator operating

state when suspended to standby, for regulators implementing

suspend configuration constraints.

This will be one of the same strings reported by

the "state" attribute.

Where I checked that all regulators are disabled!

This again should be work closely with the hardware engineer to get more directions!

The PMIC is require for power on sequence.