**New Block Thermistors**

**Adaptation**

**PDR**

**(Embedded SW)**

Written by: Slava Chuhovich

February 2012

Table of Contents

[1. Background 3](#_Toc317774425)

[2. Overview 3](#_Toc317774426)

[3. Software Changes 4](#_Toc317774427)

[4. UI Changes 4](#_Toc317774428)

# Background

In order to comply with a new Triplex print-block mechanical structure and new, additional head filling thermistors, embedded software modifications should be performed in order to fully support new Material Replacement, Shutdown and Head Optimization wizards.

New print-block structure introduced 6 filling thermistors operation scenarios, which require a convenient way in the embedded software infrastructure and code to switch from one thermistors operation mode to another to be able relatively easy replace material in required wizards.

# Overview

New print-block has 6 thermistors: 4 low thermistors and 2 high thermistors. System material filling and operation modes are defined by grouping currently operational thermistors:

Thermistor indexes

5

4

3

2

1

0

Chamber M1

Chamber S

Chamber M3

Chamber M2

In embedded software, there are 2 main parameters, which are responsible for this behavior. Their values must be dependent. ThermistorsOperationMode gives developer an easier tool to change the operation mode by enumeration and send the updated ActiveThermistors bit-mask value to hardware.

There are 4 filling modes: High – both “big” chambers are “flooded”. Low – each pair of tanks is filling it’s corresponding chamber. And combinations: Support High Model Low and vice versa.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ActiveThermistors** | | | | | | **ThermistorsOperationMode** | |
| 0 | 1 | 2 | 3 | 4 | 5 | enum | mode |
| 0 | 0 | 0 | 0 | 1 | 1 | 0 | High |
| 1 | 1 | 1 | 1 | 0 | 0 | 1 | Low |
| 0 | 1 | 1 | 0 | 1 | 0 | 2 | SH ML |
| 1 | 0 | 0 | 1 | 0 | 1 | 3 | SL MH |

# Software Changes

* ThermistorOperationMode – Must have 4 enumeration values (malfunctioned – fixed)
* Add “on change” observer to ThermistorOperationMode, to automatically change to a corresponding value ActiveThermistors parameter, which is sent to OCB.
* CContainerBase::UpdateChambers – Add an option to update chambers with new materials without setting thermistors operation mode to default. Needed for HOW.
* ResinFlowUtils – Make sure that all conversion methods are working properly (pumps by tanks / by chambers / by thermistors / by pipes etc. …)
* Implement new frontend message and logic to reflect filling operation mode changes on the application GUI.

# UI Changes

The maintenance screen of the main application UI are automatically reflected dependent on Thermistors Operation Mode:



Low or SL MH modes

High or SH ML modes