**KBA-170812191927**

**[Description]:**How to force QC3.0 to output 9V?

**[Platform]:** MSM8998, SDM845, SDM660, SDM670

**[Solution]:**

Now PM660/PMI8998 firstly enter continuous mode to authenticate QC3.0 when insert the HVDCP. So when inserting QC3.0 charger, it will stay at continuous mode and we have to force 9V by writing the cmd register(0x1343 SCHG_USB_CMD_HVDCP_2) if we want it work at 9V.

The fact is the voltage of usbin failed to raise up when force 9V in continuous mode. New voltage request to HVDCP need DM pull-down for a delay time Tglitch_dm_low, and this delay time need at least 1ms as the internal document described. But in continuous mode DP=0.6V, DM=3.3V, which doesn't meet the request.
So **before force 9V, need to force 5V first**, DM be configured as High-Z, and pull-down by the charger, then the new request will be detected.

Customer can refer to below sample code and configure the property <qcom,qc3-disable> to let QC3.0 work at 9V.

```
@qpnp-smb2.c
struct smb_dt_props {
bool hvdcp_disable;
++ bool qc3_disable;
bool auto_recharge_soc;
int wd_bark_time;
};

static int smb2_parse_dt(struct smb2 *chip)
{
...................
chip->dt.hvdcp_disable = of_property_read_bool(node,
"qcom,hvdcp-disable");
++ chip->dt.qc3_disable = of_property_read_bool(node,
++ "qcom,qc3-disable");
of_property_read_u32(node, "qcom,chg-inhibit-threshold-mv",

static int smb2_init_hw(struct smb2 *chip)
{
......................
vote(chg->hvdcp_disable_votable_indirect, DEFAULT_VOTER,
chip->dt.hvdcp_disable, 0);
```

```
++ vote(chg->hvdcp_hw_inov_dis_votable, DEFAULT_VOTER,
++ chip->dt.qc3_disable, 0); //if qc3 is disabled, inov is not allowed.
vote(chg->pd_disallowed_votable_indirect, CC_DETACHED_VOTER,

@smb-lib.h
struct smb_charger {
.................
++ bool qc3_force_9v;
};

@smb-lib.c
static void smblib_handle_apsd_done(struct smb_charger *chg, bool rising)
{
......................
++if((apsd_result->bit & (QC_3P0_BIT | QC_2P0_BIT)) //detect QC
++ && (get_client_vote_locked(chg->hvdcp_hw_inov_dis_votable, DEFAULT_VOTER)) //qc3_
disable is true
++ && (!get_effective_result(chg->hvdcp_disable_votable_indirect)) //hvdcp is enabled
++ && (!chg->qc3_force_9v) ){ //not yet force 9v
++ chg->qc3_force_9v = true;
++ smblib_dbg(chg, PR_INTERRUPT, "Force output 9V\n");
++ msleep(300);
++ rc = smblib_masked_write(chg, CMD_HVDCP_2_REG,
++ FORCE_5V_BIT, FORCE_5V_BIT); //Before request 9V, need to force 5V first.
++ msleep(300);
++ rc = smblib_masked_write(chg, CMD_HVDCP_2_REG,
++ FORCE_9V_BIT, FORCE_9V_BIT);
++}
smblib_dbg(chg, PR_INTERRUPT, "IRQ: apsd-done rising; %s detected\n",
apsd_result->name);
}

static void smblib_handle_typec_removal(struct smb_charger *chg)
{
...................
++ chg->qc3_force_9v = false;


static int smblib_hvdcp_enable_vote_callback(struct votable *votable,
...................
-- if (!hvdcp_enable)
```

```
++ if (!hvdcp_enable || get_client_vote_locked(chg->hvdcp_hw_inov_dis_votable,
DEFAULT_VOTER)) //if qc3 is disabled then no auth
val = HVDCP_EN_BIT;
```