松下電器　ヘルスケア社　御中

X100 タイプ LCD 時刻表示ズレ問題に関する報告書

1. 結論

リアルタイムクロック(RTC)精度に誤差が生じた事で LCD 表示時刻ズレ問題が発生しておりました。
Linux Open Source のリアルタイムクロック(RTC)コード部へ初期化処理及び、64Hz カウンタの桁上がり時の特定レジスタ値参照回避処理を追加したことで現象改善が確認されました。

2. 現象

SH7727 内蔵リアルタイムクロック(RTC)内 64Hz カウンタ(R64CNT)レジスタの 64Hz ビットのトグル周期に不安定な周期があり、この周期にタイマユニット(TMU)で計測したタイマカウンタ(TCNT)レジスタ値にばらつきが生じた為、Linux カーネルモジュールクロック 10ms タイマに狂いを生じさせていました。

3. 原因

3.1. RTC 初期化未実施

Linux OpenSource である RTC コード部 (time.c ソース内) にリアルタイムクロック (RTC) の初期化関数が行われていなかった為、リアルタイムクロックが正常動作していなかった。

3.2. 64Hz カウンタレジスタ(R64CNT)の桁上がり仕様

SH7727 ハードウェアマニュアル「16.2.15 章」に下記記述があります。

ハードウェアマニュアル抜粋。
「・ビット 7：桁上げフラグ (CF)
桁上げが発生したことを示すフラグです。このフラグが 1 にセットされた場合、(1) 秒カウンタ桁上げ、または、(2) 64Hz カウンタ桁上げ時の読み出しが発生したことを示し、この時点で読み出したカウントレジスタの値は、保証されません。再度の読み出しが必要です。」

上記の説明から、この桁上がりタイミングを基準としたタイマユニット(TMU) のタイマカウンタ(TCNT)レジスタで計測したカウンタ値が Linux カーネルモジュールクロック 10ms タイマの狂いをまれに生じさせていました。

4. 対策案

下記に示す 2 つ対策案のどちらか 1 つを採用される事により、正確な Linux カーネルモジュールクロックである 10ms タイマを生成することが出来、且つ LCD 時刻表示などの精度向上を実現することが可能となります。

4.1. パラメータ指定

Linux カーネルモジュールクロック自動計算を使用せず、パラメータ指定のみにする。

4.2. リアルタイムクロック(RTC)初期化処理追加及び桁上がり時のレジスタ参照回避

Linux カーネルモジュールクロック自動計算を使用する場合、リアルタイムクロック(RTC)の初期化を行い、且つ、64Hz カウンタ(R64CNT)レジスタ値を参照する際には、カウンタ値 H'7F と H'00(64Hz カウンタレジスタの桁上がりタイミング時の値)以外の値を使用してタイマユニット(TMU)のタイマカウンタ(TCNT)レジスタ値を参照する。

5. 検証手順 (過程) について

以下、これまで弊社で検証した手順(過程) について御説明させて頂きます。

5.1. Linux カーネルモジュールクロックの設定手順

Linux カーネルは CPU のクロックソースが不明である為、基準となる Linux カーネルクロック 10ms タイマの設定値を SH7727 リアルタイムクロック(RTC)とタイマユニット(TMU)を用いて自動計算します。

10ms タイマの元となる Interval 値を RTC128Hz 期間(7.8125ms)中の TMU のカウンタ数値で求めます。

・リアルタイムクロック(RTC)を使用し一定期間内に変化するタイマユニット(TMU)の TCNT 値を算出(Interval 算出)
・Linux カーネルは Interval 値からタイマユニット(TMU)のレジスタ設定を行って 10ms タイマを生成する

5.2. 原因 1 (リアルタイムクロック初期化未実施)

リアルタイムクロック (RTC) の初期化について検証した手順 (過程) を御説明いたします。

5.2.1. 実機での現象確認

現象を確認するために「計測 Start-Stop 間」期間を計測できるよう、汎用ポート PTK0 に計測 Start 時 High、Stop 時 Low となるデバッグ文を入れてオシロスコープで時間を測定しました。 図 1 にオシロスコープの波形を示します。

図 1 から RTC128(7.8125ms)を超えている (TCNT 値が期待値より多くカウントされている) 事が分かりました。

Linux カーネルはリアルタイムクロック (RTC) を使用した一定期間内(計測 Start-Stop 間)の TCNT 値の差を求めますが、毎回 TCNT 値が違う値となっており、期待値 (48,000 カウント) とならない事象が発生していることを確認しました。

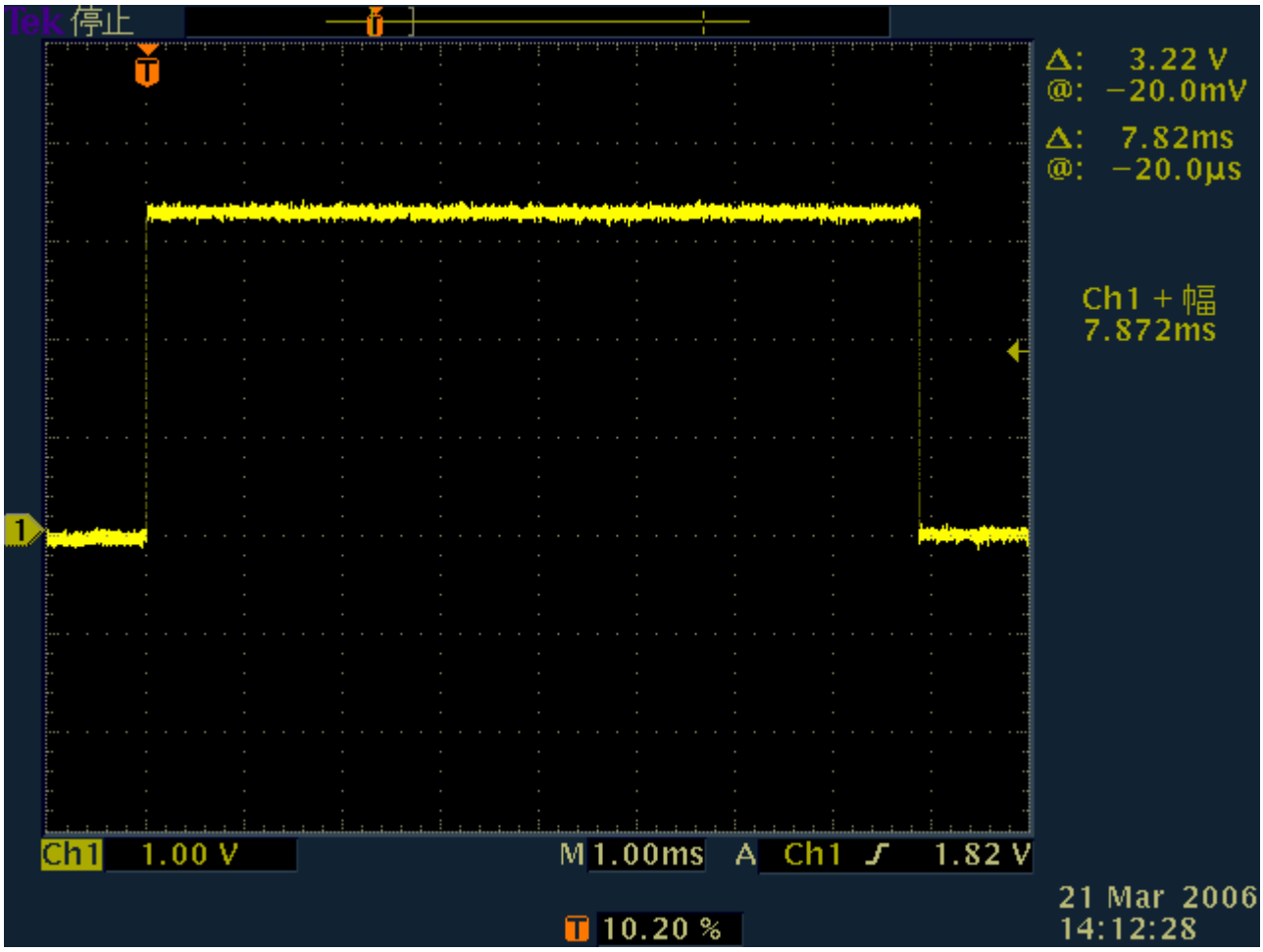

図 1 「計測 Start-Stop 間」期間波形

5.2.2. 仮説

時間ずれの原因となる仮説（下記 2 点）をたてて検証を実施しました。

①外部要因

タイマユニット（TMU）の TCNT レジスタ値を読み出す際に何らかの外部要因が発生。
（=RTC 計測 Stop 点が後にずれて TCNT 値が増加した。）

②リアルタイムクロック(RTC)精度

何らかの要因でリアルタイムクロック（RTC）の R64CNT レジスタ精度に問題が発生する。

5.2.2.1. 「①外部要因」検証

Interval 値を求める「計測 Start-Stop 間」期間(1/128Hz=7.8125ms)中に外部割込み、外部ウェイトが発生していないか、外部割込み発生時は汎用ポート PTK1(CH2)を High、外部ウェイト端子(CH3)にて調査しました。
下記にオシロスコープの波形を示す。

CH1 = 「計測 Start-Stop 間」期間(PTK0)
CH2 = 外部割込み発生信号(PTK1)
CH3 = 外部 WAIT 端子

「計測 Start-Stop 間」期間中に外部割込みが発生した場合は CH2 が High、外部ウェイトが発生した場合は CH3 が Low となるはずですが、図 2 からも分かるように外部割り込み、外部ウェイトは発生していませんでした。
（例外割込みについては、SH7727 用 E10A エミュレータを使用し例外割込みが発生していないことも確認しました。）

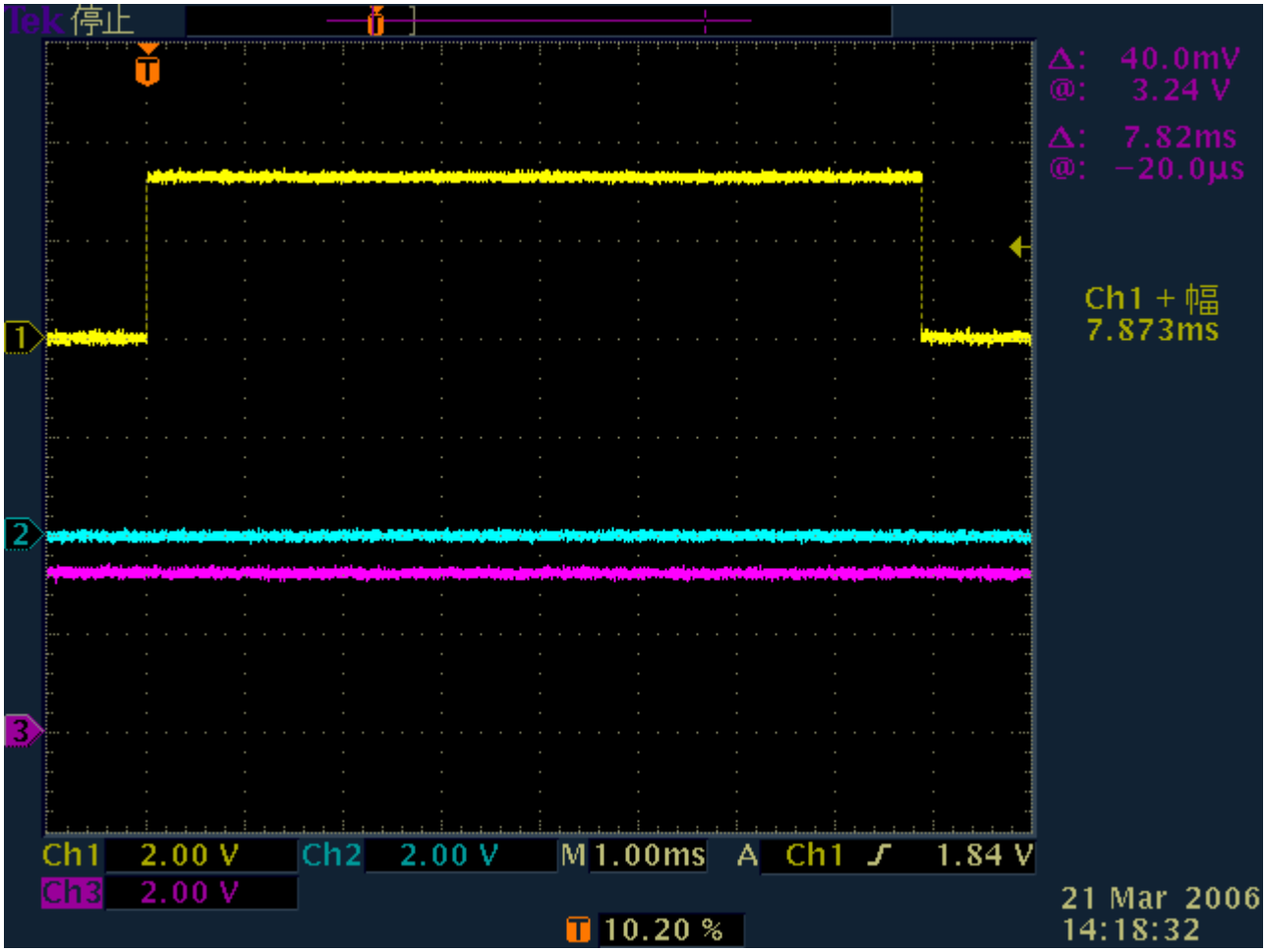

図 2 外部要因波形

5.2.2.2. 「②リアルタイムクロック(RTC)精度」検証

リアルタイムクロック(RTC)精度を測定するにはリアルタイムクロック(RTC)の R64CNT レジスタの 64Hz ビット(ビット0)を使用し、このビットのトグルで信号を反転させるデバッグ文を入れ 128Hz クロック波形となるように汎用ポートPTK1(CH2)に出力させました。

下記にオシロスコープの波形を示します。

図 3 よりトリガポイントの High から Low 期間にずれが確認されました。

(RTC 精度起因で「計測 Start-Stop 間」期間が長くなり、TCNT 値を多く得ていたことが分かります。)

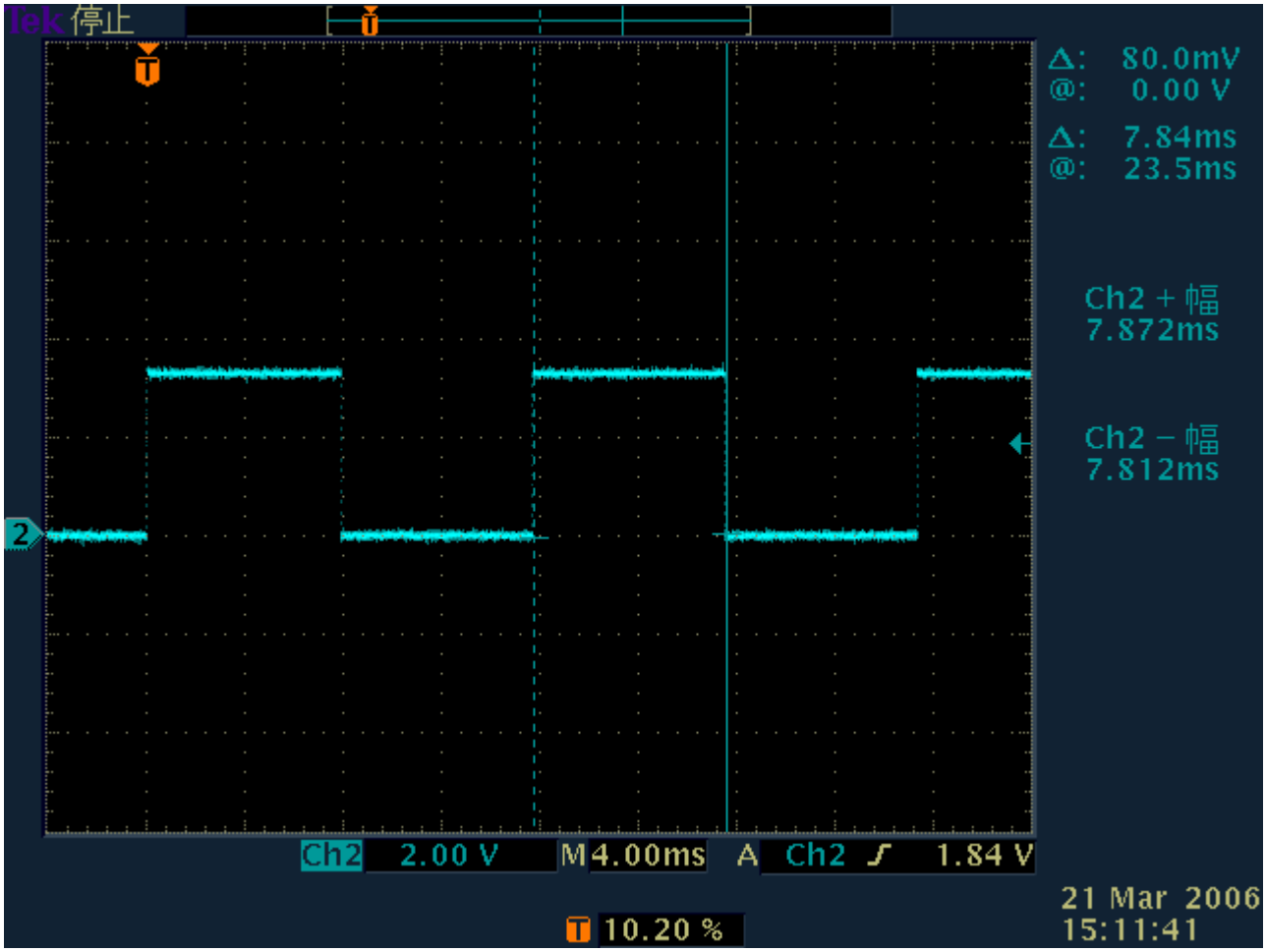

図 3 R64CNT 64Hz ビットトグル波形

5.2.3. 原因

リアルタイムクロック (RTC) 精度に誤差を生じさせた箇所を特定するため、Linux Open Source でのリアルタイムクロック (RTC) 該当箇所のソースレビューを行った結果、Interval 値を求める「計測 Start-Stop 間」期間の前にリアルタイムクロック (RTC) 初期化処理が行われていない事を突き止めました。
(今回の使用方法で顕在化した現象と、その Linux Open Source コード該当箇所 {起因コード} と考えられます。)

5.2.4. 対策

リアルタイムクロック (RTC) 初期設定処理を Linux Open Source へ追加した結果、正確な R64CNT 64Hz ビットトグルを確認する事が出来ました。

Time.c ソース内 get_timer_frequency 関数に下記 RTC 初期化処理を追加
(詳細は別添 time.c ソース参照して下さい。)

```
/* RTC reset */
ctrl_outb(0x02, RCR2);        /* reset */
udelay(150);
ctrl_outb(0x09, RCR2);        /* start */
udelay(150);
```

図 4 に改善した R64CNT 64Hz ビットトグル波形を、図 5 に「計測 Start-Stop 間」期間波形を示します。

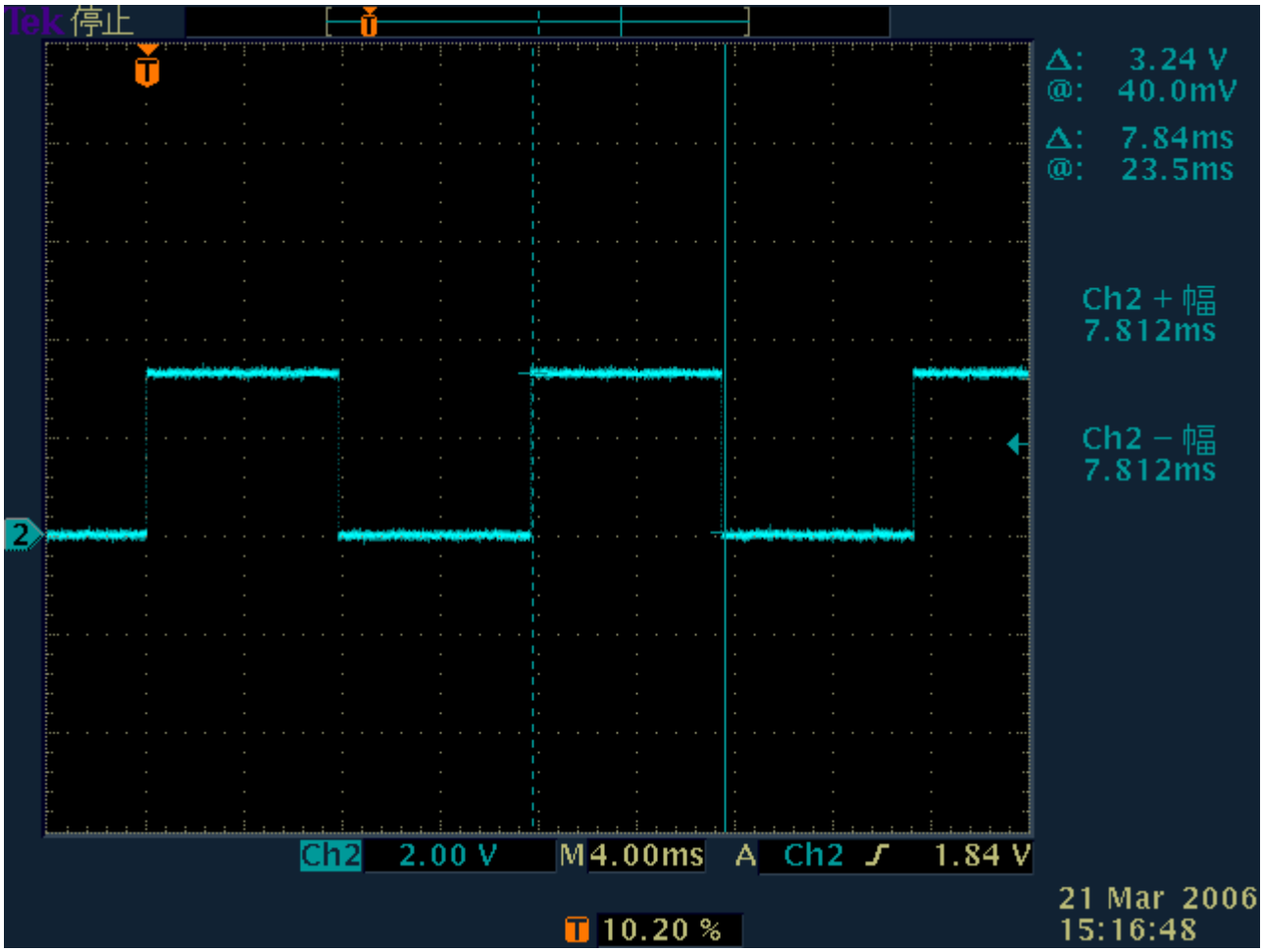

図 4 R64CNT 64Hz ビットトグル波形

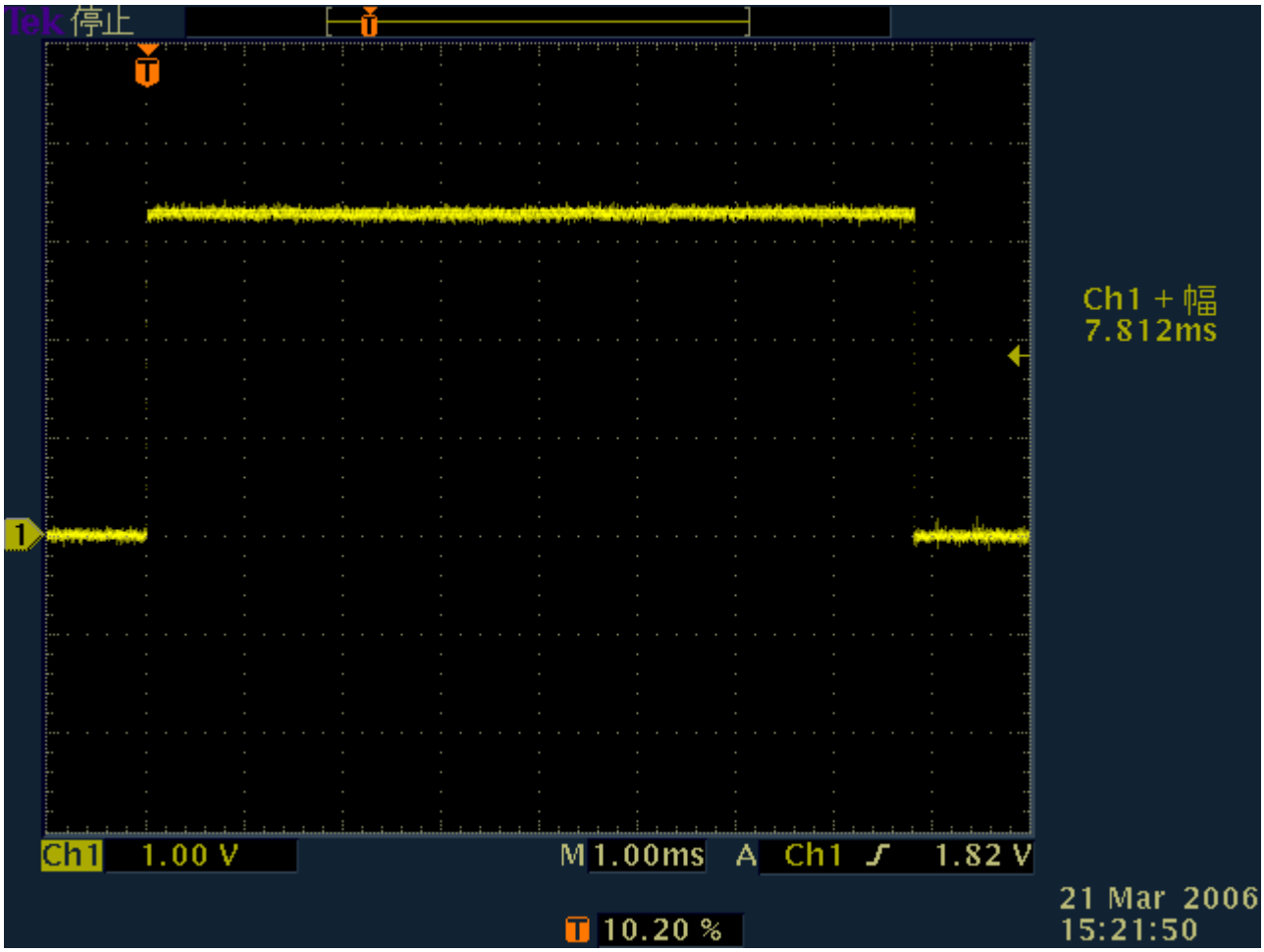

図 5 「計測 Start-Stop 間」期間波形

5.2.5. 結果

Linux Open Source のリアルタイムクロック (RTC) コード部に初期化処理を追加することで現象改善 (時計表示はストップウォッチ計測で1時間を経過誤差 1s 以内) を確認できました。

5.3. 原因 2 (64Hz カウンタレジスタについて)

松下電器ヘルスケア社長本様からのご指摘メール (2006/03/28 (火) 16:26) **にもありますように弊社においても追跡調査を行いました。 以下にリアルタイムクロック** (RTC) **の** 64Hz **カウンタ** (R64CNT) **レジスタについて検証した手順** (過程) **を御説明いたします。**

5.3.1. 現象

Linux Open Source **のリアルタイムクロック** (RTC) **コード部に初期化処理を追加したにもかかわらず、まれに** Linux **カーネルクロック** 10ms **タイマを設定する** Interval **値が期待値**(=TCNT **レジスタ値** 48000) **とならない事象が発生する事が分かりました。**

実機での現象を確認する為に秒カウンタクロック波形と 64Hz **カウンタクロック波形を生成させ波形を測定しました。図** 6 **にオシロスコープの波形を示します。**

図 6 **からリアルタイムクロック** (RTC) **の** 64Hz **カウンタレジスタ** (R64CNT) **のカウンタ桁上がり前後に** 64Hz **ビットトグル** (64Hz) **のデューティ比が必ずズレる事象が発生していることを確認しました。**

(但し、桁上がり前後以外は正常動作します。)

CH1 = **秒カウンタレジスタ** RSECCNT **波形**(PTK0)
CH2 = 64Hz **カウンタ** R64CNT bit0 **波形**(PTK1)

(R64CNT **の桁上がり直前**) = **レジスタ値** H'7F → 7.752ms --- (a)
(R64CNT **の桁上がり直後**) = **レジスタ値** H'00 → 7.872ms --- (b)
但し、上記(a)+(b)**は** 15.624ms

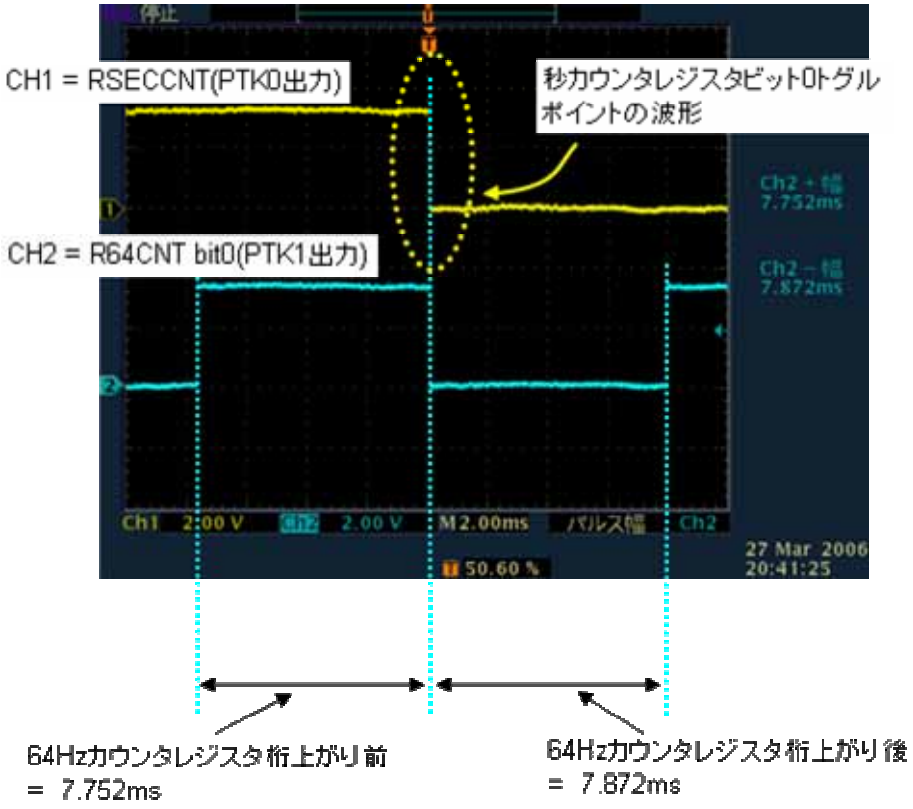

図 6 実測波形

5.3.2. 原因

SH7727 ハードウェアマニュアル「16.2.15 章」に下記記述があります。

SH7727 ハードウェアマニュアルから抜粋。
「・ビット 7：桁上げフラグ (CF)
桁上げが発生したことを示すフラグです。このフラグが 1 にセットされた場合、(1) 秒カウンタ桁上げ、または、(2) 64Hz カウンタ桁上げ時の読み出しが発生したことを示し、この時点で読み出したカウントレジスタの値は、保証されません。再度の読み出しが必要です。」
以上抜粋。

上記の説明から、64Hz カウンタ (R64CNT) レジスタの桁上がり時には正確な 1/128sec を計測できない事がタイマユニット(TMU) のタイマカウンタ(TCNT)レジスタで計測したカウンタ値が Linux カーネルモジュールクロック 10ms タイマの狂いをまれに生じさせる原因となっていました。

5.3.3. 対策

64Hz カウンタ(R64CNT)レジスタ値を参照する際には、カウンタ値 H'7F と H'00(64Hz カウンタレジスタの桁上がり時)以外の値を使用する処理を Linux Open Source へ追加した結果、正確な Interval 値を算出することを確認しました。
図 7 にカウンタ値 H'7F と H'00 以外の値を参照する対策を施した Interval 値算出のフローチャートを図 8 に RTC 取得のフローチャートを示します。(Interval 値計測フローを説明する為に便宜上 RTC 取得フローを説明致します。)

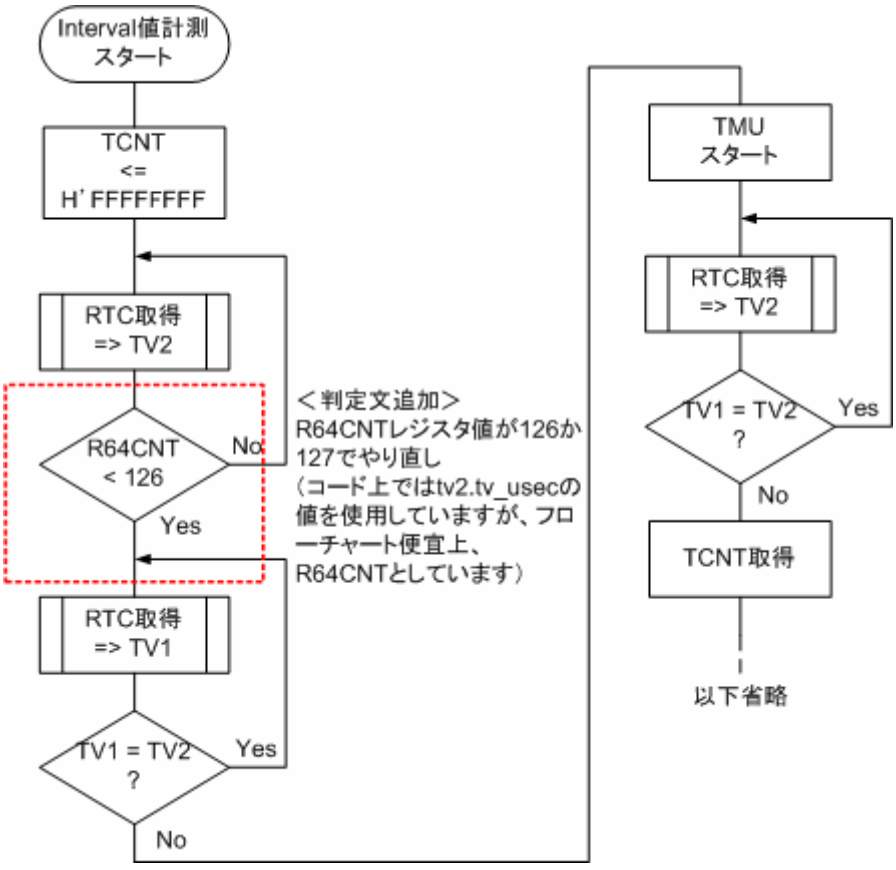

図 7 Interval 値計測フロー

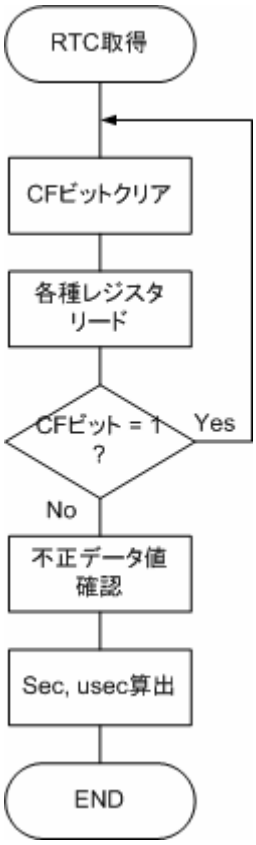

図 8　RTC 取得フロー

5.3.4. 結果

Linux Open Source のリアルタイムクロック (RTC) コード部に 64Hz カウンタレジスタの桁上がり時以外の値を使用する処理を追加することで現象改善 (時計表示はストップウォッチ計測で1時間を経過誤差 1s 以内) を確認できました。

5.3.5. 波形生成方法

5.3.5.1. 秒波形生成

秒カウンタレジスタ RSECCNT のビット 0 のトグルをソフトウェアで判定し、前回と値が違っていたら汎用ポート PTK0 に Low か High を出力させる処理を追加し、出力されたポート状態をオシロスコープにて波形を測定しました。図 9 に波形生成のフローチャートを示します。

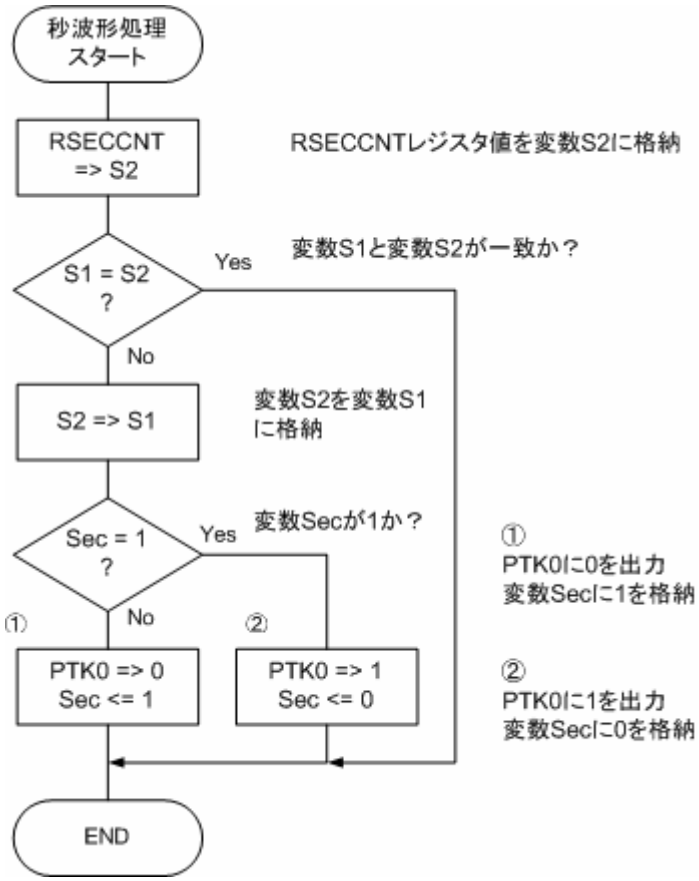

図 9 秒生成フローチャート

5.3.5.2. 64Hz 波形生成

64Hz カウンタレジスタ R64CNT のビット 0 (64Hz) のトグルをソフトウェアで判定し、前回と値が違っていたら汎用ポート PTK1 に LOW か HIGH を出力させる処理を追加し出力されたポート状態をオシロスコープにて波形を測定しました。図 10 に波形生成のフローチャートを示します。

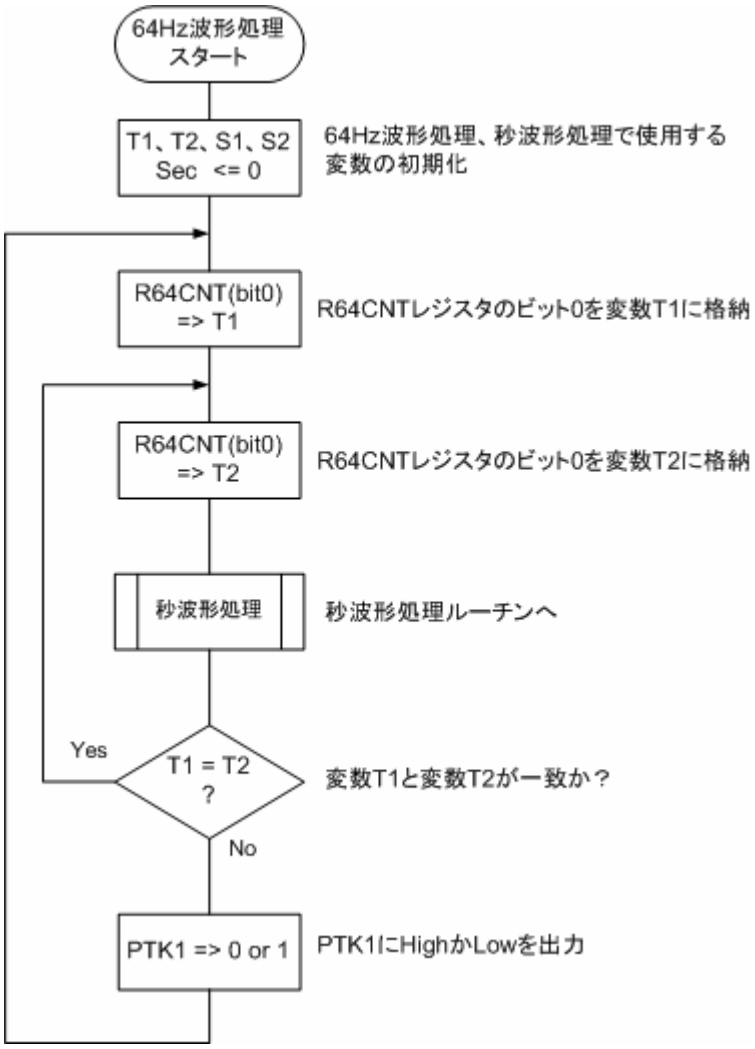

図 10 64Hz 波形生成フローチャート

5.4. 付録

5.4.1. LCD 表示時間ズレ確率

LCD 表示時間ズレの発生する確率を処理毎に表で示します。

5.4.1.1. リアルタイムクロック (RTC) 初期化処理を行わなかった場合

表 1 初期化未実施確率

項目	時間誤差	確率
1	0.811%	42%
2	0.002%	9%
3	0.769%	24%
4	0.044%	2%
5	0.771%	23%

5.4.1.2. リアルタイムクロック (RTC) 初期化処理を行った場合

表 2 初期化実施確率

項目	時間誤差	確率
1	0.000%	44%
2	0.040%	21%
3	-0.002%	35%

5.4.1.3. リアルタイムクロック (RTC) 初期化処理＋64Hz カウンタ(R64CNT)レジスタの桁上がり時以外の値を使用する処理

表 3 初期化実施＋桁上がり時以外の値使用の確率

項目	時間誤差	確率
1	0.000%	46%
2	0.040%	6%
3	0.002%	37%
4	0.039%	9%
5	0.044%	2%

以上