

Application Note
GR47/GR48 Current consumption

CE

The product described in this manual conforms to the TTE directive 91/263/EEC and EMC directive 89/336/EEC. The product fulfils the requirements according to ETS 300 342-1.

The information contained in this document is the proprietary information of **Sony Ericsson Mobile Communications**. The contents are confidential and any disclosure to persons other than the officers, employees, agents or subcontractors of the owner or licensee of this document, without the prior written consent of **Sony Ericsson Mobile Communications**, is strictly prohibited.

Further, no portion of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording, without the prior written consent of **Sony Ericsson Mobile Communications**, the copyright holder.

First edition (August 2003)

Sony Ericsson Mobile Communications publishes this manual without making any warranty as to the content contained herein. Further **Sony Ericsson Mobile Communications** reserves the right to make modifications, additions and deletions to this manual due to typographical errors, inaccurate information, or improvements to programs and/or equipment at any time and without notice. Such changes will, nevertheless be incorporated into new editions of this manual.

All rights reserved.

© **Sony Ericsson Mobile Communications**., 2003

Contents

1	INTRODUCTION.....	4
2	TEST METHOD	5
3	RESULTS.....	6
3.1	NETWORK ATTACHED	6
3.2	CONTROLLER MODE	7
3.3	NETWORK ATTACHED WITH EMBEDDED SCRIPT	8
4	SUMMARY.....	9
	APPENDIX I - CONTROLLER MODE TEST SCRIPT	11
	APPENDIX II - NORMAL MODE TEST SCRIPT.....	12

1 Introduction

When not in stand-by, the GR47 has several modes of operation which will continuously drain supply current. It is desirable to know the current consumption figures for each of these modes of operation.

This report documents the current consumption figures for typical modes of operation.

2 Test Method

A resistance of 21.4Ω was placed in series with the GR47 power supply (+3.8V) and voltage measurements were taken across the resistor every 0.5 seconds over a period of a few minutes. Care was taken to ensure that there was no significant additional current drain from active general purpose ports or non-essential circuitry.

The modes of operation measured were:

MODE	DESCRIPTION
Network Attached and Radio Enabled	When the module has just started or after recent network activity the radio remains enabled for a period of time.
Network Attached and Radio Idle	After a period of time without network activity the radio is put into idle mode (but activated for short bursts during network updates)
Controller Mode with Script Active	Controller mode is an M2Mpower operational state in which all non-essential features are disabled.
Controller Mode with Script Idle	The embedded script is placed in an 'idle' mode whenever an internal delay is called [using intrinsic function either <code>dlyms()</code> or <code>dlys()</code>].
Controller Mode with Script Stopped	When the script comes to its natural conclusion at the end of function <code>main()</code> it is considered 'stopped'.
Network Attached with Embedded Script Running	When not in controller mode, the radio performs its normal operations and may also have an embedded script performing control functions. The current consumption will vary depending on the state of the network, radio and embedded script.

The different modes of operation can be clearly identified from the charts in the following section.

3 Results

3.1 Network Attached

Chart 1 shows the module being switched on, attached to the network and then entering into 'radio idle' mode after about 1.5 minutes. The average currents during this time are:

Mode	Average	Unit
Network Attached – Active	7.962	mA
Network Attached – Idle	4.484	mA

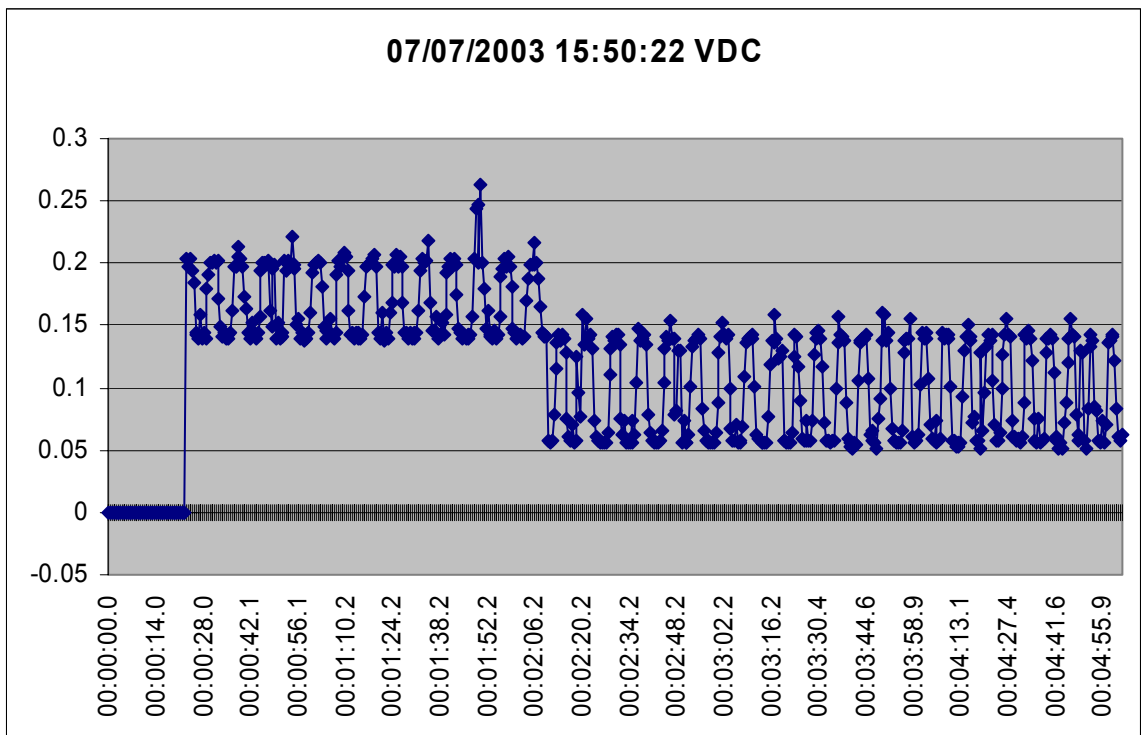


Chart 1 - Modes during Network Attached

3.2 Controller Mode

Chart 2 shows the module being switched on in controller mode. A copy of the script used may be found in Appendix I. The script performs six loops, switching between 30 seconds of 'idle' and 30 seconds of line-by-line interpreter operation, after which the script 'stops' reaching the end of function main().

The average currents during these times are:

Mode	Average	Unit
Controller Mode Idle	4.268	mA
Controller Mode Active	9.919	mA
Controller Mode Stopped	4.268	mA

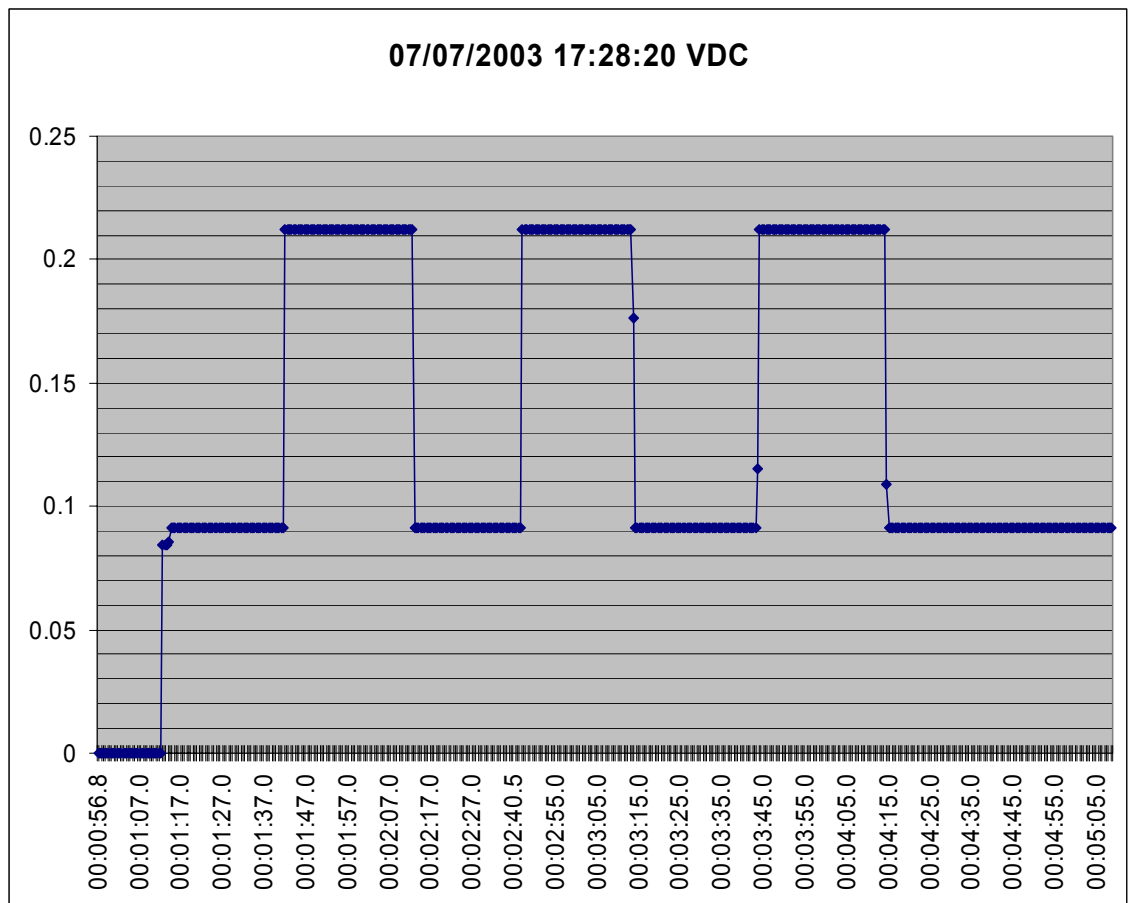


Chart 2 - Controller Mode running Embedded Script

3.3 *Network Attached with Embedded Script*

Chart 3 shows the module running a script whilst also attached to a network. After about 1.5 minutes the module enters into 'radio idle' mode. A copy of the script used may be found in Appendix II. The script performs six loops, switching between 30 seconds of 'idle' and 30 seconds of line-by-line interpreter operation, after which the script 'stops' reaching the end of function main().

The average currents during these times are:

Mode	Average	Unit
Radio Enabled – Script Active	13.544	mA
Radio Enabled – Script Idle	8.227	mA
Radio Idle – Script Active	10.140	mA
Radio Idle – Script Idle	4.618	mA
Radio Idle – Script Stopped	4.635	mA

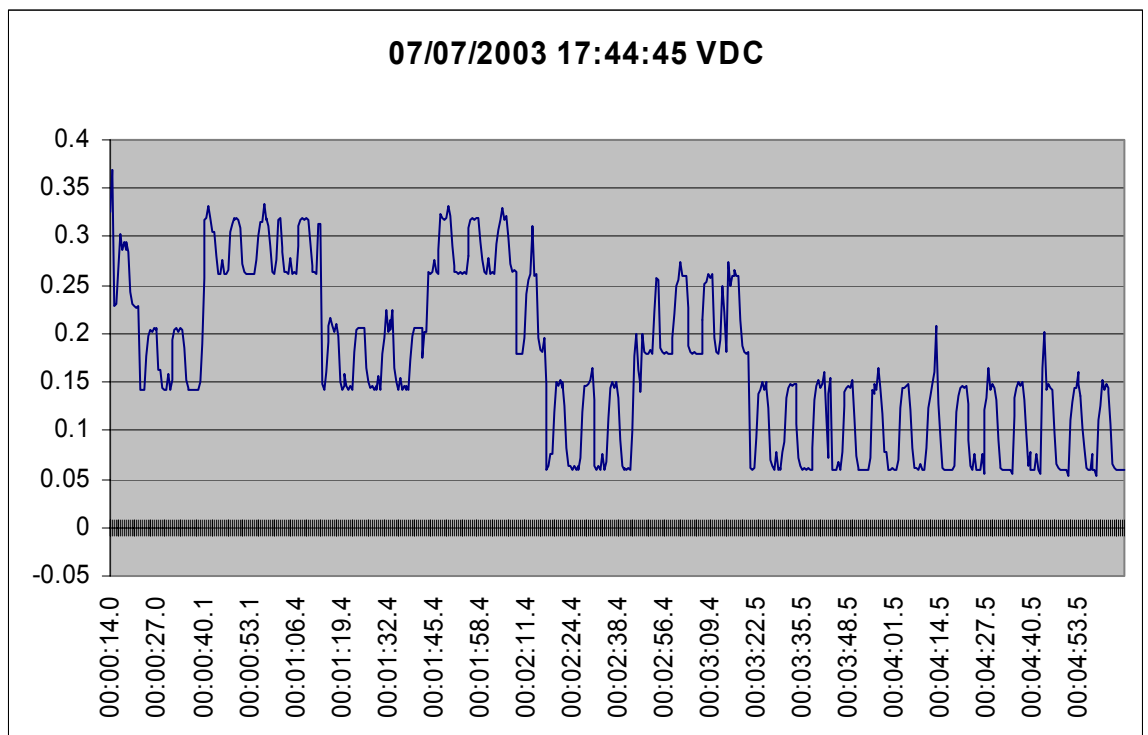


Chart 3 - Combination Network Attached with Script Running

4 Summary

During a powered-on state the minimum current consumption is achieved when in controller mode with the interpreter idle (i.e. using one of the intrinsic delay functions). The current measured under these conditions is typically 4.3mA.

There is a current attributed to each of the modes of operation of the module identified by the step changes in the charts. To estimate the typical current consumption of a module the relevant current can be added to the module's minimum current of 4.3mA.

The modes and currents identified are provided in the following table with examples relating to the operating modes found in Chart 3.

The current was measured by taking voltage readings across a 1k resistor in series with the module supply line. i.e. 50mV = 50uA.

In the shutdown state the following microprocessor and PSU circuits are powered:

- 1.8V LP regulator for RTC (Real Time Clock)
- RTC backup charging
- RTC processor and alarm functions
- PSU startup monitor, interrupt and Reset

The RTC charges the backup capacitor to 1.8V through an 18k series resistor. Worst case current draw when backup capacitor is fully discharged = 100uA. The developer's kit uses a 0.3F backup capacitor which takes approximately 8-10 hours to charge fully.

Once the backup capacitor is fully charged (or if RTC backup is not used) the current drawn by the GR47 during power-down state is 50uA +/- 10uA (across the supply range 3.4V-4.0V).

	Mode	Current
A	Minimum Current (Controller Mode and Script Idle)	4.3 mA
B	Script Active	5.6 mA
C	Radio Idle	0.3 mA
D	Radio Enabled	3.7 mA

Example Operating Modes	Using	Calculated	Measured
Controller mode – Script idle	A	4.3 mA	4.27 mA
Controller mode – script active	A+B	9.9 mA	9.92 mA
Radio Enabled – Script Active	A+B+D	13.6 mA	13.54 mA
Radio Enabled – Script Idle	A+D	8.0 mA	8.23 mA
Radio Idle – Script Active	A+B+C	10.2 mA	10.14 mA
Radio Idle – Script Idle	A+C	4.6 mA	4.62 mA
Radio Idle – Script Stopped	A+C	4.6 mA	4.63 mA

APPENDIX I - Controller Mode Test Script

```
int  APPS_INCONTROLLERMODE = 11;

/* main */

main ()
{
int  loop;
int  temp;
int  loop2;

    if (gtf(APPS_INCONTROLLERMODE) == 0)
    {
        printf("\n*TL* Switching to controller mode.\n");
        rst(1);
    }

/* Put rest of Controller Mode code here ! */
    loop2 = 0;
    for (loop2=1; loop2<4; loop2++)
    {
        printf("\n*TL* Begin 30sec Idle (loop %d)\n", loop2);
        dlys(30);
        printf("\n*TL* Now 30sec Active loop\n");
        loop = 0;
        for (loop=1; loop<7000; loop++)
        {
            temp = loop;
        }
        printf("\n*TL* End loop %d\n", loop2);
    }
    printf("\n*TL* End Embedded App\n");
}
```

Appendix II - Normal Mode Test Script

```
int  APPS_INCONTROLLERMODE = 11;

/* main */

main ()
{
int  loop;
int  temp;
int  loop2;

/*  if (gtf(APPS_INCONTROLLERMODE) == 0)
    {
        printf("\n*TL* Switching to controller mode.\n");
        rst(1);
    }*/

/* Put rest of Controller Mode code here ! */
loop2 = 0;
for (loop2=1; loop2<4; loop2++)
{
    printf("\n*TL* Begin 30sec Idle (loop %d)\n", loop2);
    dlys(30);
    printf("\n*TL* Now 30sec Active loop\n");
    loop = 0;
    for (loop=1; loop<7000; loop++)
    {
        temp = loop;
    }
    printf("\n*TL* End loop %d\n", loop2);
}
printf("\n*TL* End Embedded App\n");
}
}
```