Embedded in Life
An IXYS Company

White Paper

ZMOTION® Detection Lens and Pyro Sensor Configuration Guide

Sonia Daley, Senior Product Engineer

WP001808-1214

Copyright ©2014 Zilog® Inc. All rights reserved.
www.zilog.com

Revision History

Each instance of Revision History reflects a change to this document from its previous revision. For more details, refer to the corresponding pages linked within the table below.

Date	Revision Level	Description	Page
Dec 2014	06	Updated for content and diagrams throughout; WP0018-SC01 source code also updated.	All
Apr 2011	05	Updated Recommended Intrusion API Settings table.	6
Mar 2011	04	Updated to include recommended API settings for the Intrusion Detection MCU.	6
Jan 2011	03	Updated to include two additional Nicera lenses.	n/a
Nov 2010	02	Updated for new Zilog/IXYS logo	All
Aug 2010	01	Initial Release	All

Introduction

Zilog's ZMOTION® Product Family is a series of high-performance microcontrollers with integrated PIR motion detection software. This motion detection software comprises the PIR Engine and runs in the background while control and status of the Engine is accessed through a software Application Programming Interface (API). The PIR Engine can be configured to operate with a variety of lenses and pyroelectric sensors by initializing the API with appropriate values.

Zilog provides these API initialization values for a wide range of lenses via API_INIT header files that are to be included with your application software.

This document provides ZMOTION PIR Engine API configuration settings optimized and validated by Zilog for a variety of lens and pyroelectric sensor combinations. These API settings can be used as-is or serve as a starting point for further optimization, depending on the specific application and environmental requirements. Walk test results of the lens and sensor combination are included for reference.

Test Configuration and Test Method

The API settings and walk test results are split in to two categories in this document:

Detection and Control. Lenses and pyroelectric sensors for general-purpose applications such as lighting control, HVAC, kiosks, etc.

Intrusion. Lenses and pyroelectric sensors designed specifically for intrusion/security applications.

The ZMOTION Detection & Control Development Kit (ZMOTIONL100ZCOG) is used to test the *detection and control* lens and pyroelectric sensor combinations included in this document. To learn more about this development kit, refer to the ZMOTION Detection and Control Development Kit User Manual (UM0230).

For lenses specifically targeting intrusion applications, use the ZMOTION Intrusion Development Kit (ZMOTIONS200ZCOG). To learn more about this development kit, refer to the ZMOTION Intrusion Development Kit User Manual (UM0233).

For testing purposes, each of the above kit's development boards are mounted at the lens manufacturer's recommended height, and walk tests are performed at a variety of target speeds. The `ZMOTION_Serial_Config` sample project provided with each development kit is used for the user application code.

Each lens and pyroelectric sensor combination contains its own associated configuration header file that can be included with the application code. Each file is named based on its configuration, `API_INIT_xx.h`, in which `xx` represents the numbers that identify the file, as indicated in the API File Name row in Table 1.

Recommended API Settings: Detection and Control

Table 1 lists the recommended API settings for multiple lens and sensor combinations. These values represent a balance of performance and stability for most applications and environments. Adjustments may be made to suit a particular application.

API_Init header files with these settings can be found in the ZMOTION Detection Lens and Pyro Sensor source code file, WP0018-SC01.

Table 1. Recommended Detection and Control API Settings

PIR Lens	PIR Sensor	API Init Header File Number	Sensitivity	Extended Detection Level	Frequency Response	PIR Scan Rate	Range Control	Lock Level	Window Size	Window Update Rate	Sample Size	Debounce Time	Debounce batch Size
ZNCL10IL	ZRE200GE	06	12	0	8	0	2	2	3	2	32	120	FF
ZNCL3B	ZRE200GE	07	12	0	8	0	2	2	3	2	32	120	FF
ZNCL10R	ZSBG446671	13	12	0	8	0	2	2	3	2	32	120	FF
ZNCL10R	ZRE200GE	13	12	0	8	0	2	2	3	2	32	120	FF
ZNCL10S	ZRE200GE	12	12	0	8	0	2	2	3	2	32	120	FF
ZNCL926 (Wall)	ZRE200GE	05	12	0	8	0	2	2	3	2	32	120	FF
ZNCL926 (Ceiling)	ZSBG446671	05	12	0	8	0	2	2	3	2	32	120	FF
ZNCL926 (Ceiling)	ZRE200GE	05	12	0	8	0	2	2	3	2	32	120	FF
ZNCL11	ZRE200GE	08	12	0	8	0	2	2	3	2	32	120	FF
ZCWM05GIV1 (Wall)	ZRE200GE	04	12	0	8	0	2	2	3	2	32	120	FF
ZCWM05GIV1 (Ceiling)	ZSBG446671	04	12	0	8	0	2	2	3	2	32	120	FF
CM0.5 GI V2 (Ceiling)	ZSBG446671	14	12	0	10	0	1	2	3	2	32	120	FF

Note: Settings for the ZRE200GE sensor also apply to the ZSBG323671 sensor.

Table 1. Recommended Detection and Control API Settings (Continued)

PIR Lens	PIR Sensor	API Init Header File Number	Sensitivity	Extended Detection Level	Frequency Response	PIR Scan Rate	Range Control	Lock Level	Window Size	Window Update Rate	Sample Size	Debounce Time	Debounce batch Size
CM0.5 GI V2 (Ceiling)	LHi 1128	14	12	0	10	0	1	2	3	2	32	120	FF
ZCM077GIV2	ZSBG446671	15	12	0	10	0	1	2	3	2	32	200	3F
ZCM077GIV3	ZSBG446671	02	12	0	10	0	1	2	3	2	32	200	3F
ZCM077GIV5	ZSBG446671	03	12	0	10	0	1	2	3	2	32	200	3F
ZEWA03GIV2	ZRE200GE	16	12	0	7	0	1	2	3	2	32	120	3F
ZAA09GIT1	ZRE200GE	01	12	0	8	0	2	2	3	2	32	120	3F

Note: Settings for the ZRE200GE sensor also apply to the ZSBG323671 sensor.

Recommended API Settings: Intrusion Detection

Table 2 lists the recommended API settings for multiple lens and sensor combinations specifically suited for intrusion and/or security applications. These values represent a balance of performance and stability for most applications and environments. Adjustments may be made to suit a particular application.

API_Init header files with these settings can be found in the ZMOTION Detection Lens and Pyro Sensor source code file, WP0018-SC01.

Table 2. Recommended Intrusion API Settings

PIR Lens	PIR Sensor	API Init Header File Number	Sensitivity	Extended Detection Level	Frequency Response	PIR Scan Rate	Range Control	Lock Level	Window Size	Window Update Rate	Sample Size	Debounce Time	Debounce batch Size
ZCM077GIV2	ZSBG446671	17	15	0	10	0	1	2	3	2	32	120	3F
ZCM077GIV3	ZSBG446671	18	15	0	10	0	1	2	3	2	32	120	3F
ZCM077GIV5	ZSBG446671	19	15	0	10	0	1	2	3	2	32	120	3F
ZWA12GI12V4	ZRE200GE	09	15	0	10	0	1	2	3	2	32	120	3F
ZLR12GI12V3	ZRE200GE	10	15	0	10	0	1	2	3	2	32	120	3F
ZVB12GIV1	ZRE200GE	11	16	0	10	0	2	2	3	2	32	120	FF

Note: Settings for the ZRE200GE sensor also apply to the ZSBG323671 sensor.

Walk Test Results

Figures 1 through 25 indicate the actual walk test results of each lens and sensor combination when tested with each ZMOTION Development Kit using the settings in Tables 1 and 2.

The wall-mounted lens walk tests appear on a half-circle grid with lines at 10-foot radii. Radial lines appear at 10-degree intervals from the center line.

The ceiling-mounted lens walk tests appear on a full circle grid with lines at 3-foot or 6-foot radii. Radial lines appear at 15-degree intervals.

Colored dots are used to indicate the detection capability at that specific location.

- Green dots indicate where motion was consistently detected. Target movement is one radial interval (10 or 15 degrees), through the radial line (ex. When testing the twenty-degree line, step from 15 degrees to 25 degrees).
- Purple dots indicate where motion was consistently detected when moving an extra 5-7 degrees compared to a green dot.
- Yellow dots indicate where motion was not consistently detected.

- Blue dots indicate the limit to where a partially-obscured target can be detected. This test is accomplished by holding a large section of cardboard in front of the target, covering everything but the head and shoulders. Target movement is similar to a green dot test. Not all lenses have been tested for this aspect.
- Light blue dots indicate the limit to where *micro-movement* can be detected. This test is accomplished by standing in place and moving each arm up from the side to shoulder height and back down. Not all lenses have been tested for this aspect.

The legend in each plot provides additional information about environmental conditions and target size.

Stability Tests

All lens configurations were tested for stability in a large, open-air room using the same API settings as used for the walk test results. For the Detection & Control lenses, no false motion events were recorded over a two-day (48-hour) period.

For the Intrusion lenses, no false motion events were recorded over a 14-day period.

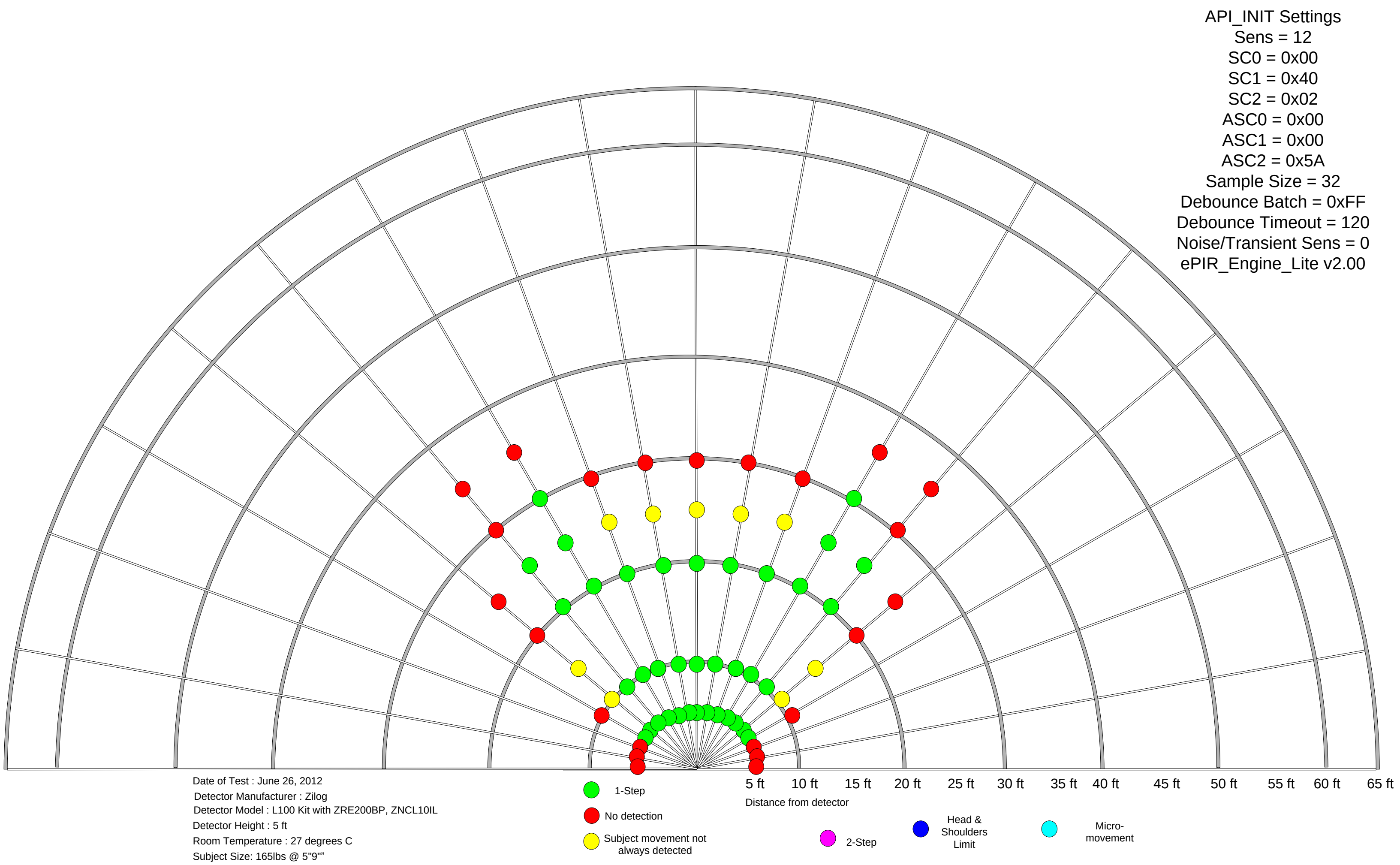

Figure 1. ZNCL10IL with ZRE200GE

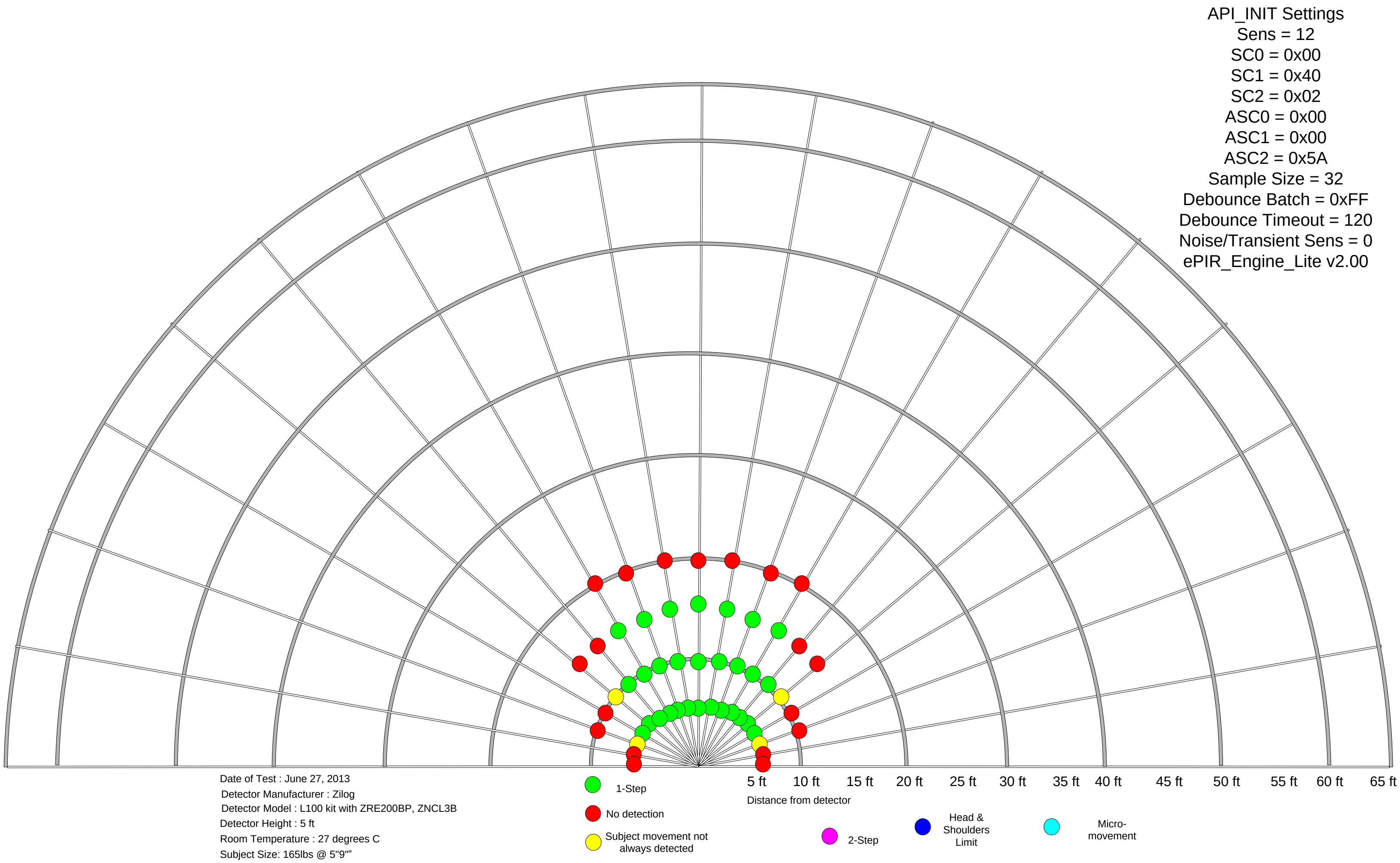

Figure 2. ZNCL3B with ZRE200GE

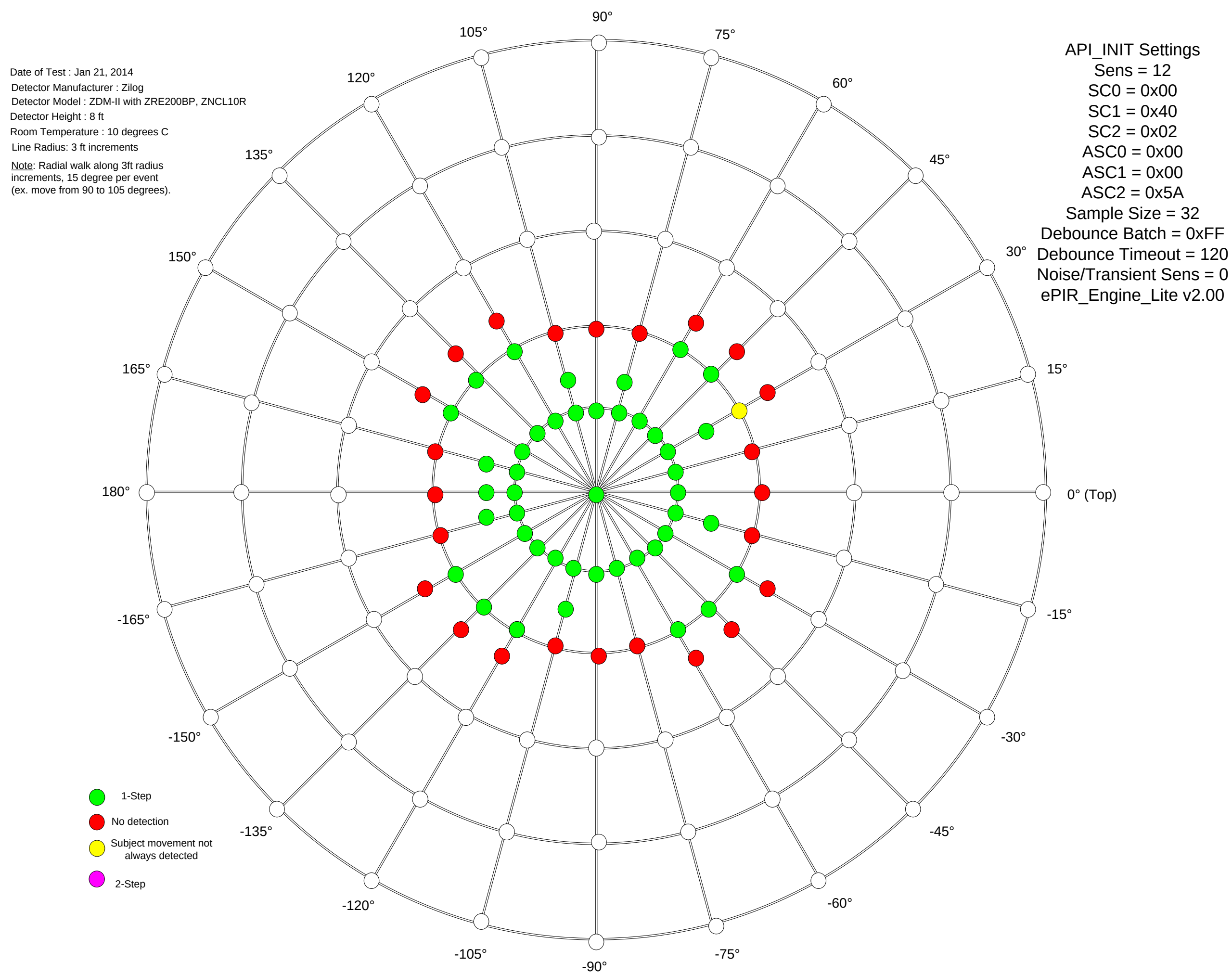

Figure 3. ZNCL10R with ZRE200GE (8-Foot Ceiling)

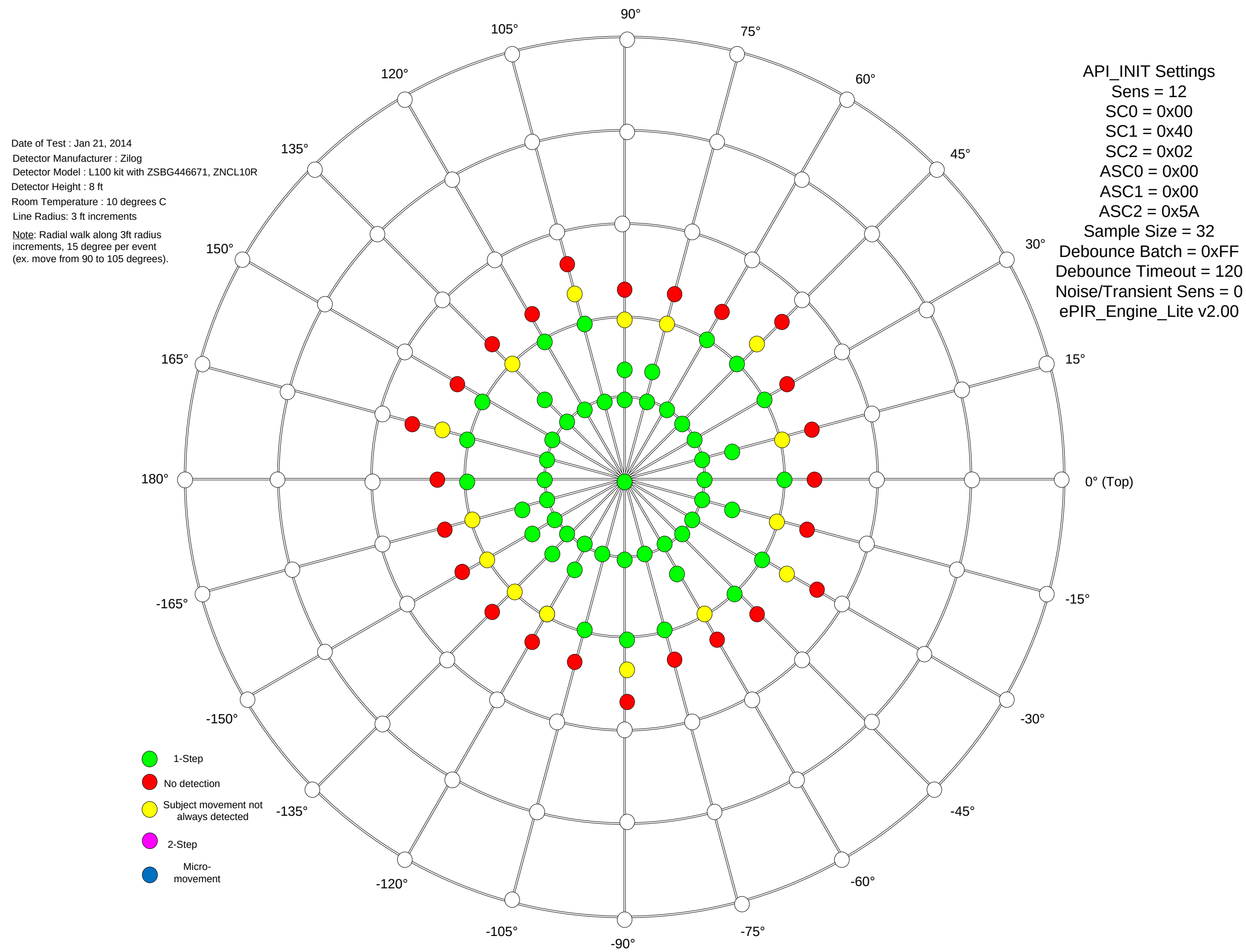

Figure 4. ZNCL10R with ZSBG446671 (8-Foot Ceiling)

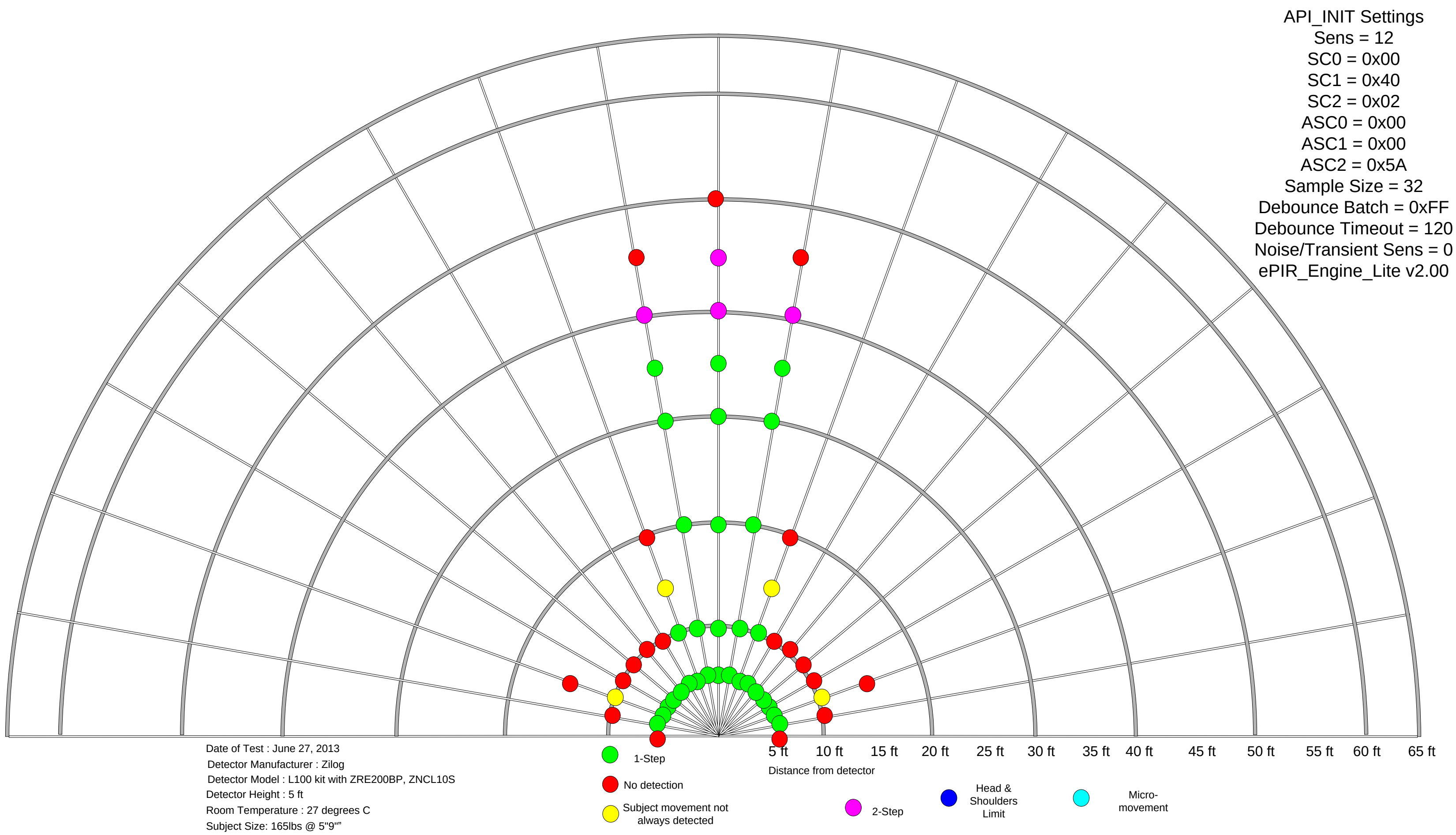

Figure 5. ZNCL10S with ZRE200GE

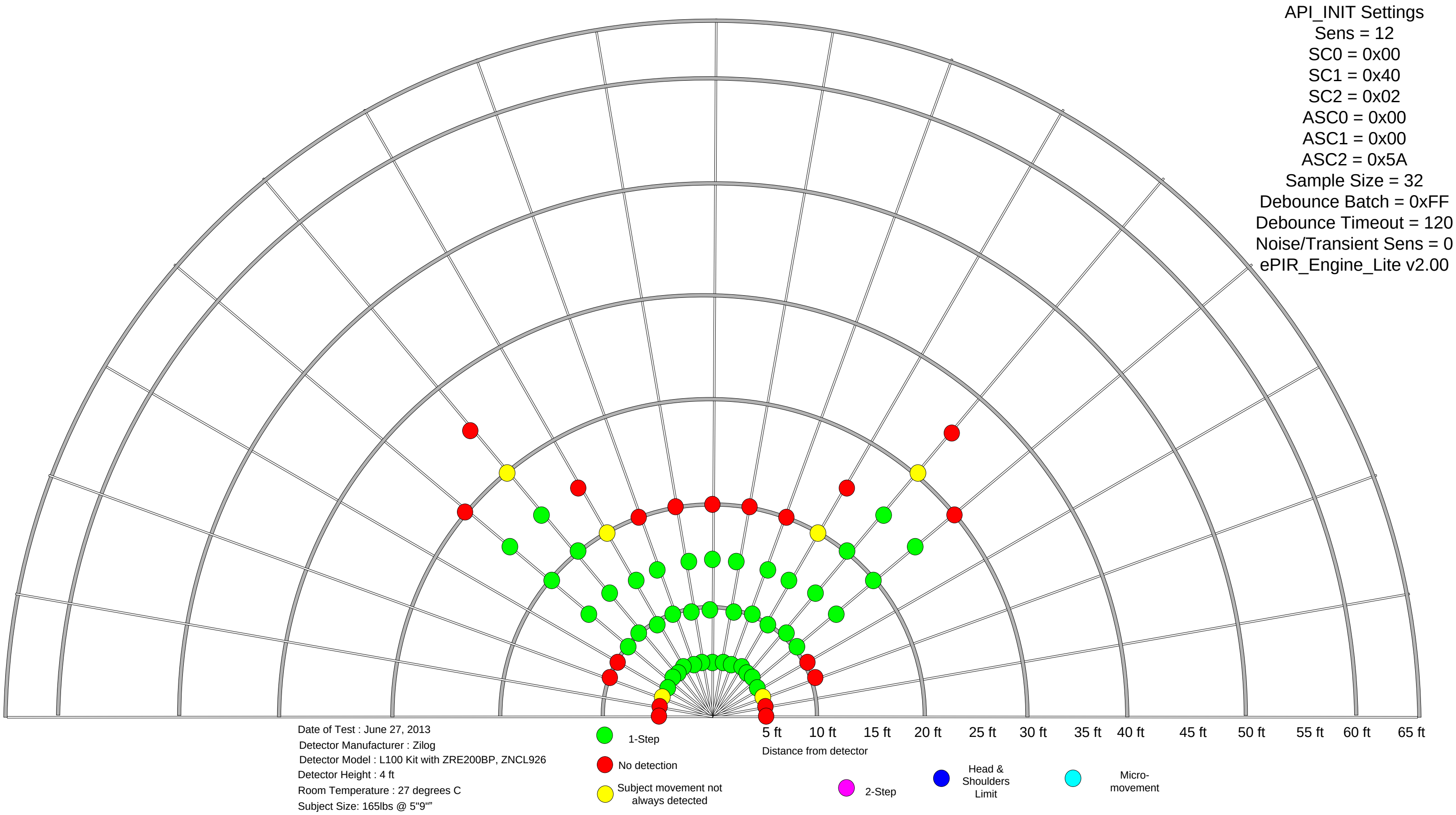

Figure 6. ZNCL926 with ZRE200GE (Wall)

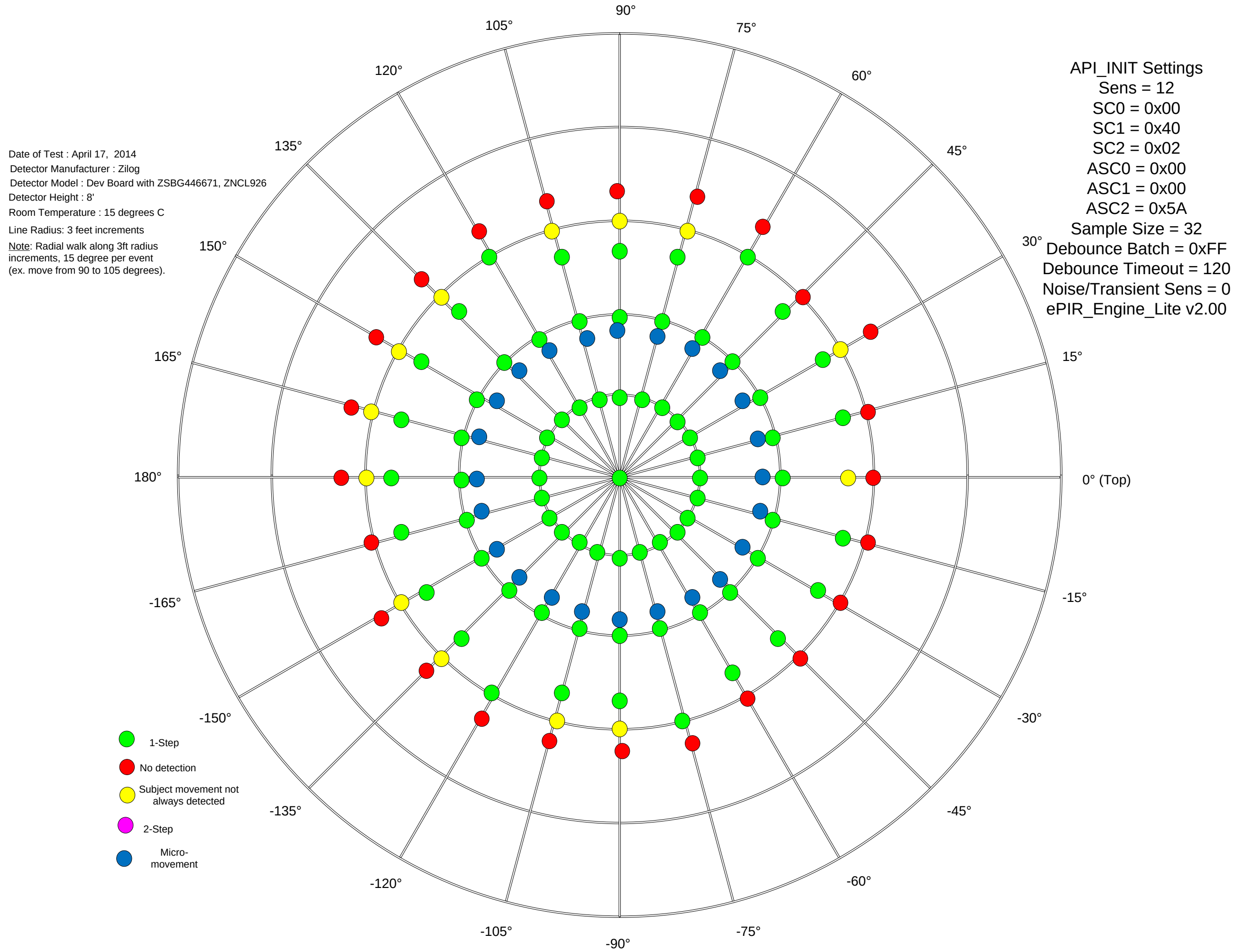

Figure 7. ZNCL926 with ZSBG446671 (8-Foot Ceiling)

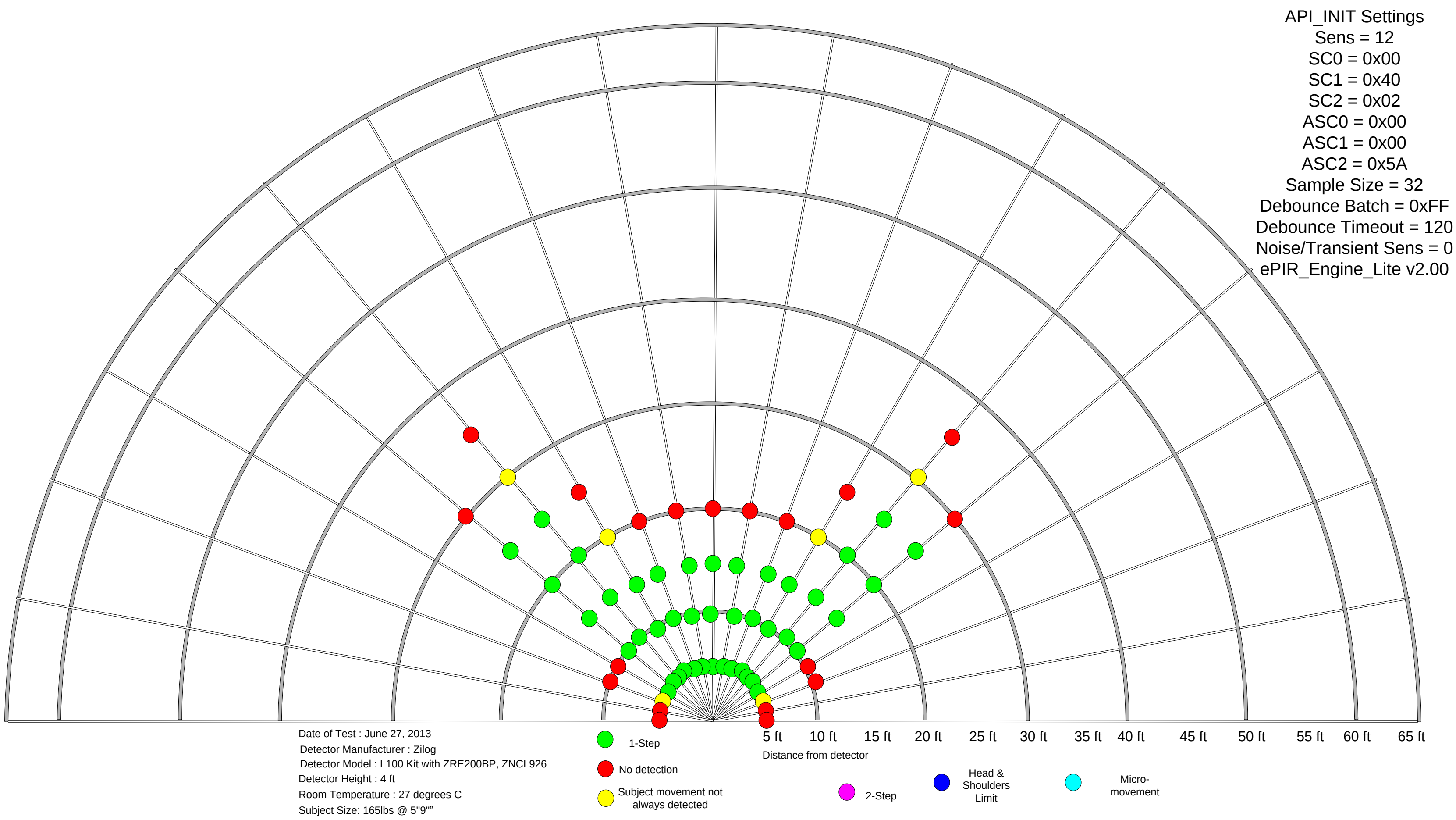

Figure 8. ZNCL926 with ZRE200GE (8-Foot Ceiling)

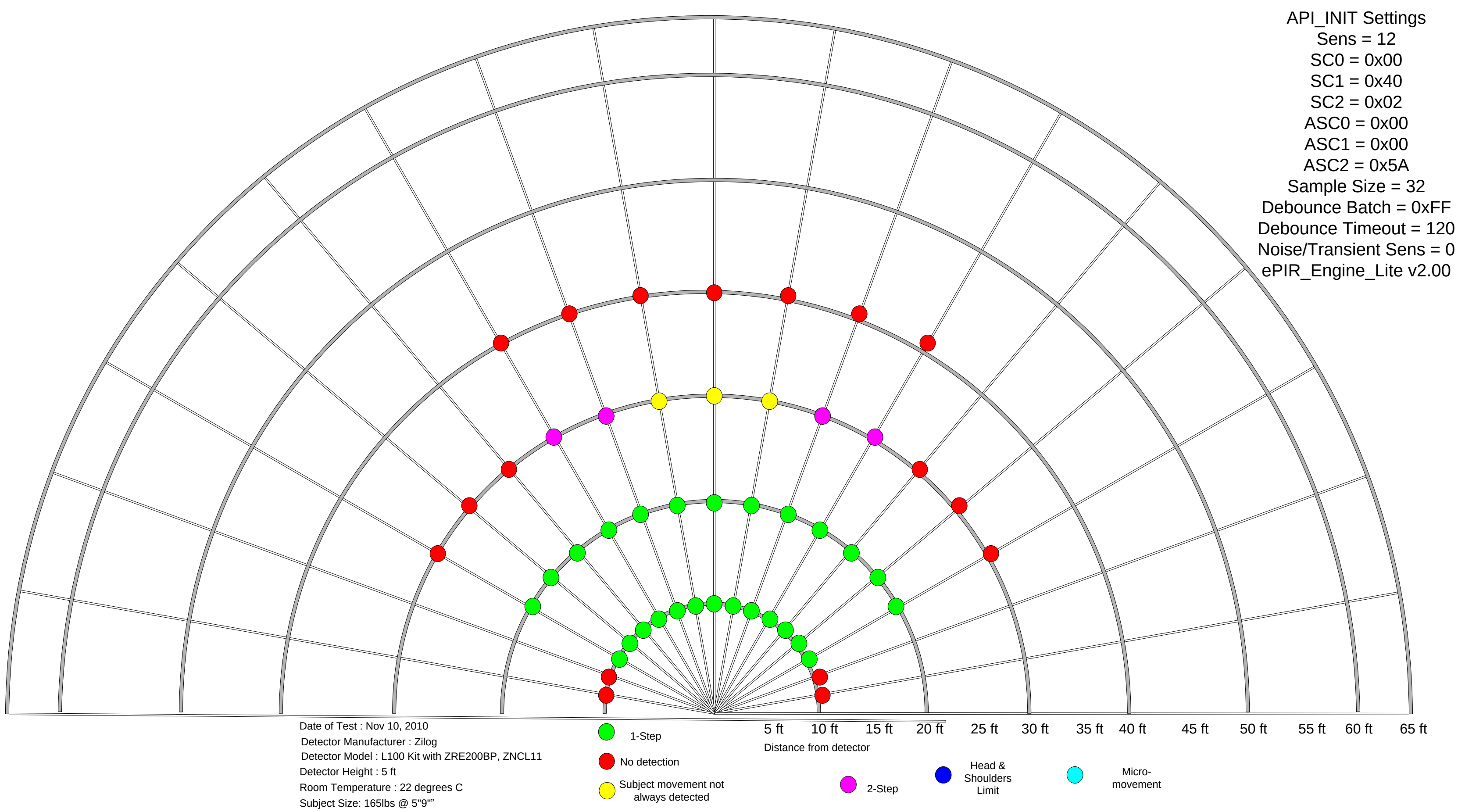

Figure 9. ZNCL11 with ZRE200GE

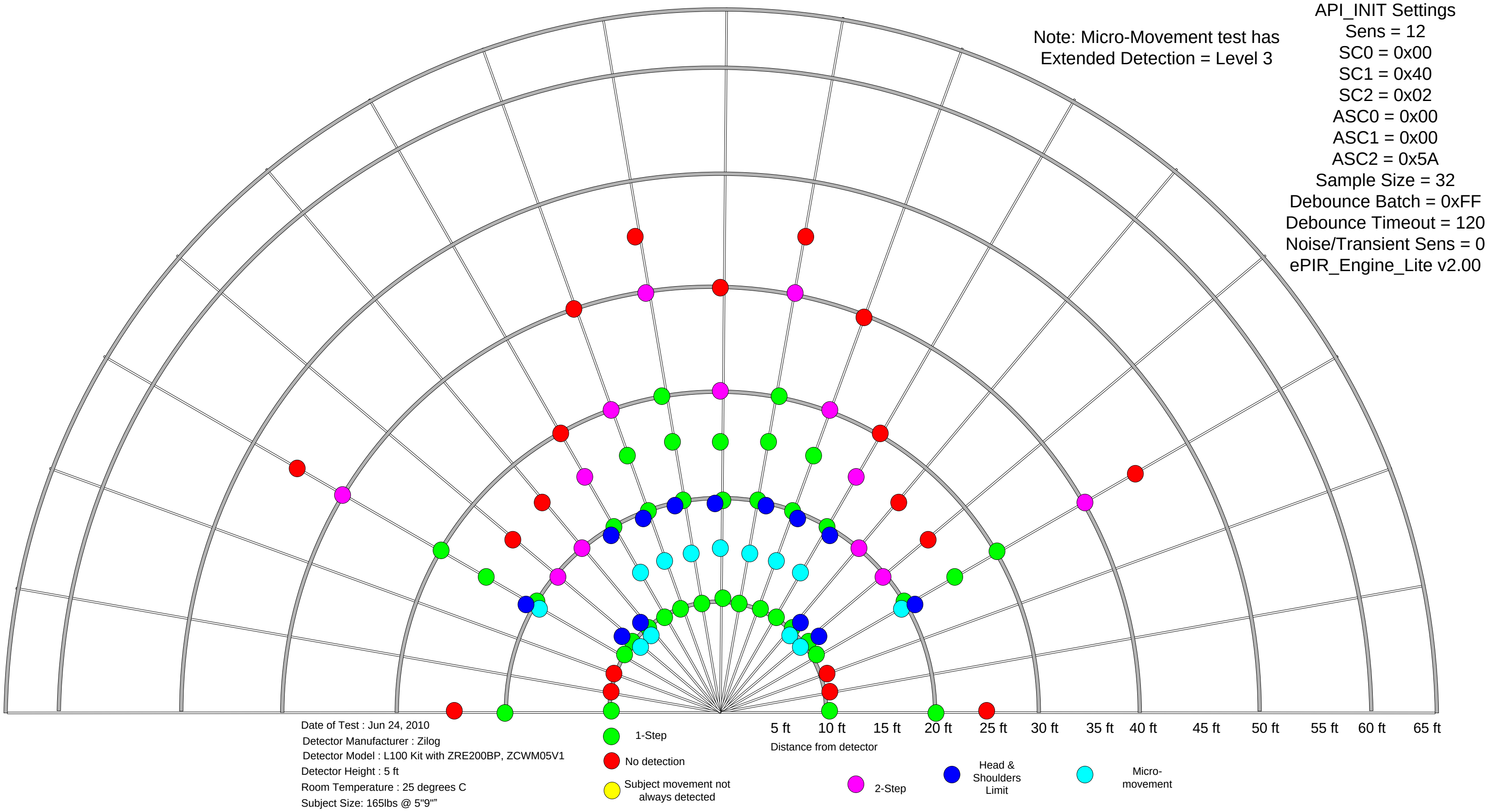

Figure 10. ZCWM05GIV1 with ZRE200GE (Wall)

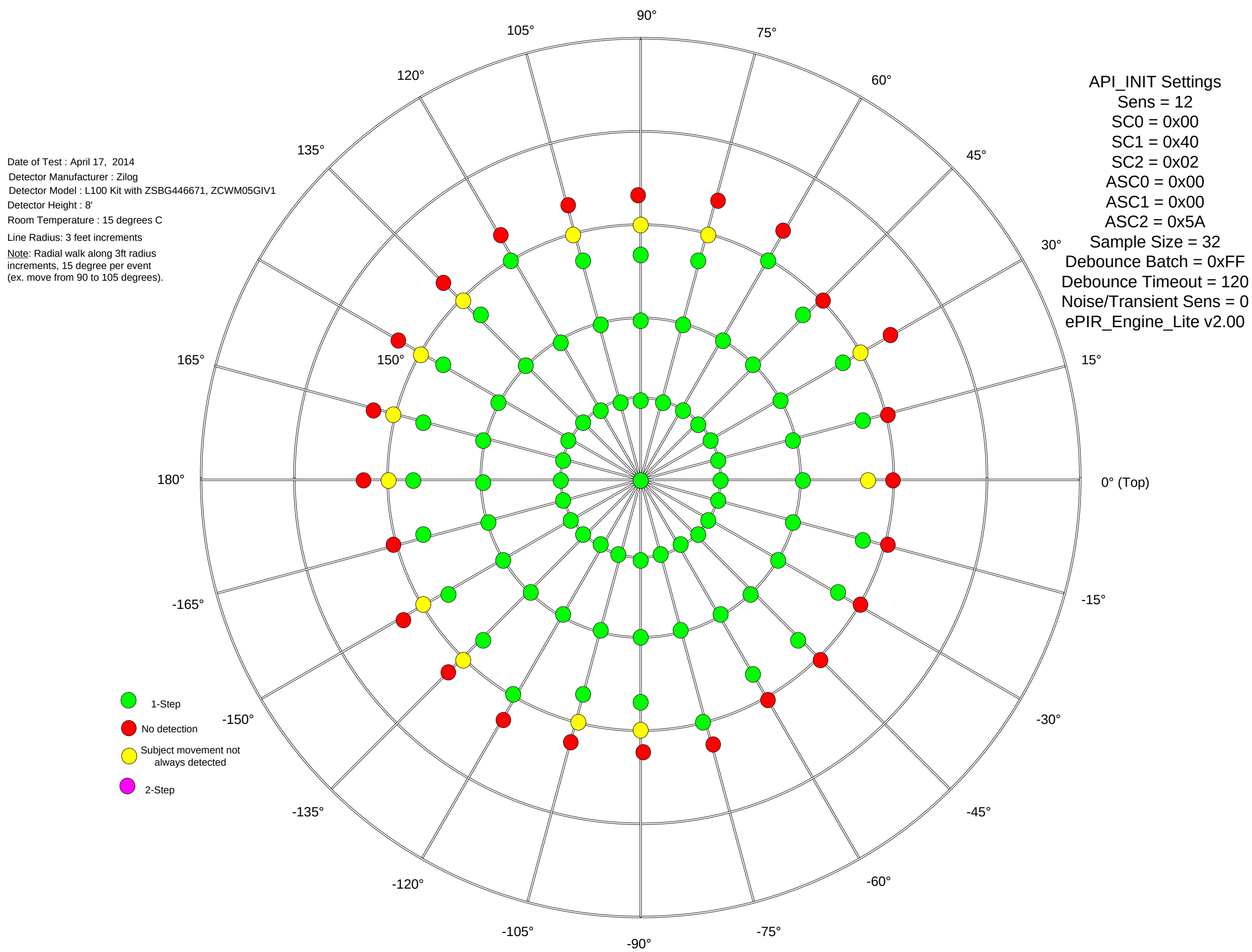

Figure 11. ZCWM05GIV1 with ZSBG446671 (8-Foot Ceiling)

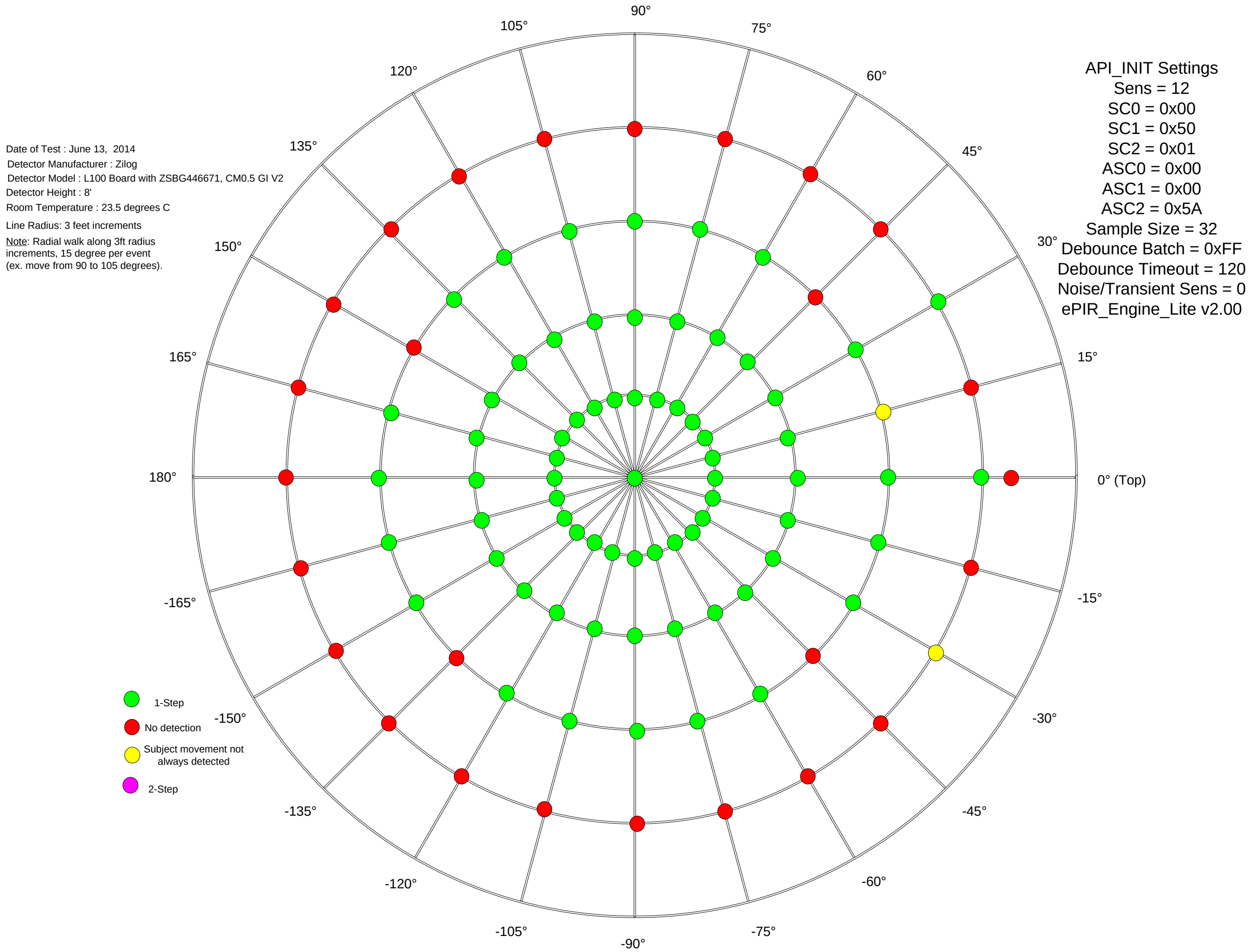

Figure 12. CM 0.5GI V2 with ZSBG446671 (8-Foot Ceiling)

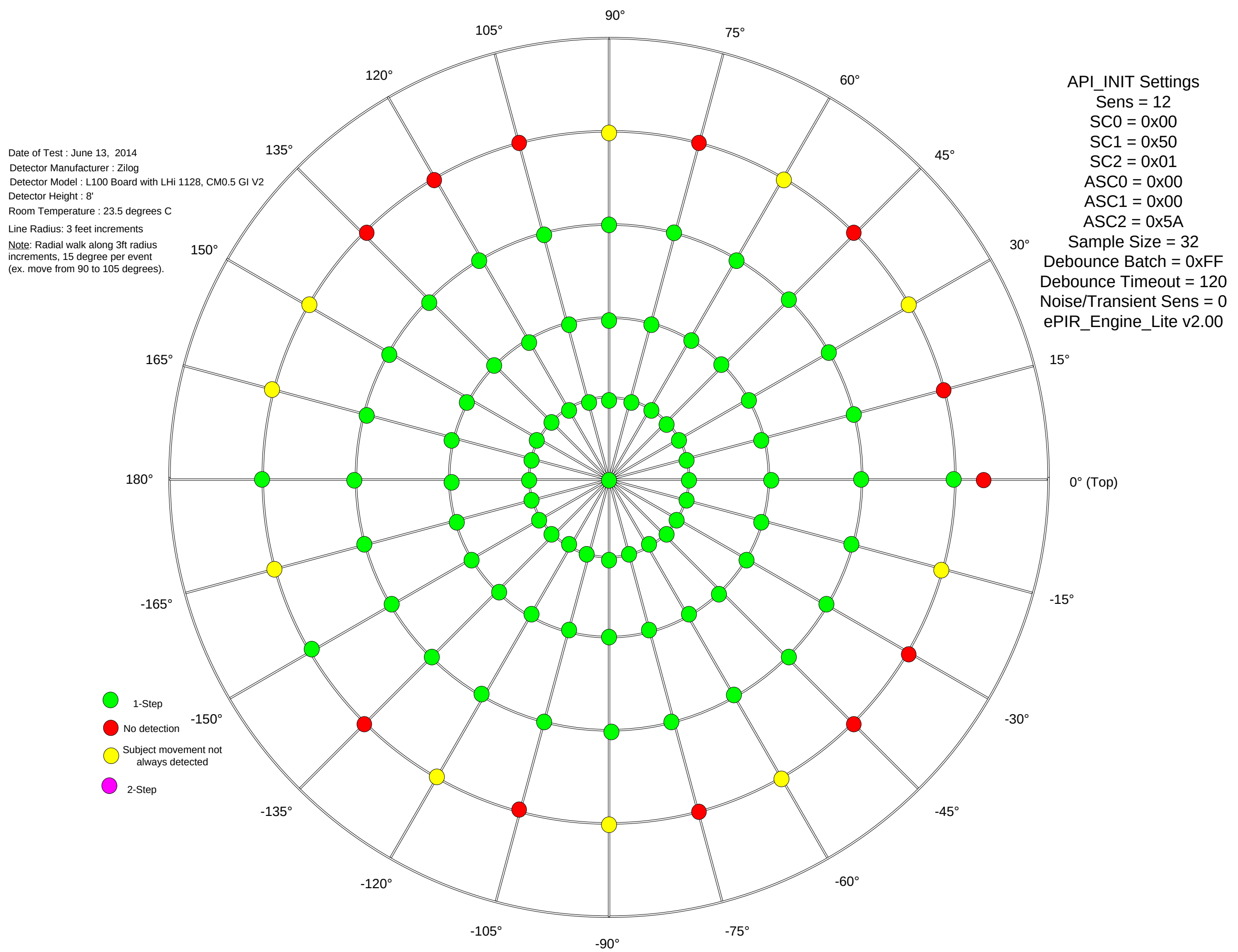

Figure 13. FTI CM 0.5 GI V2 with Excelitas LHi 1128 (8-Foot Ceiling)

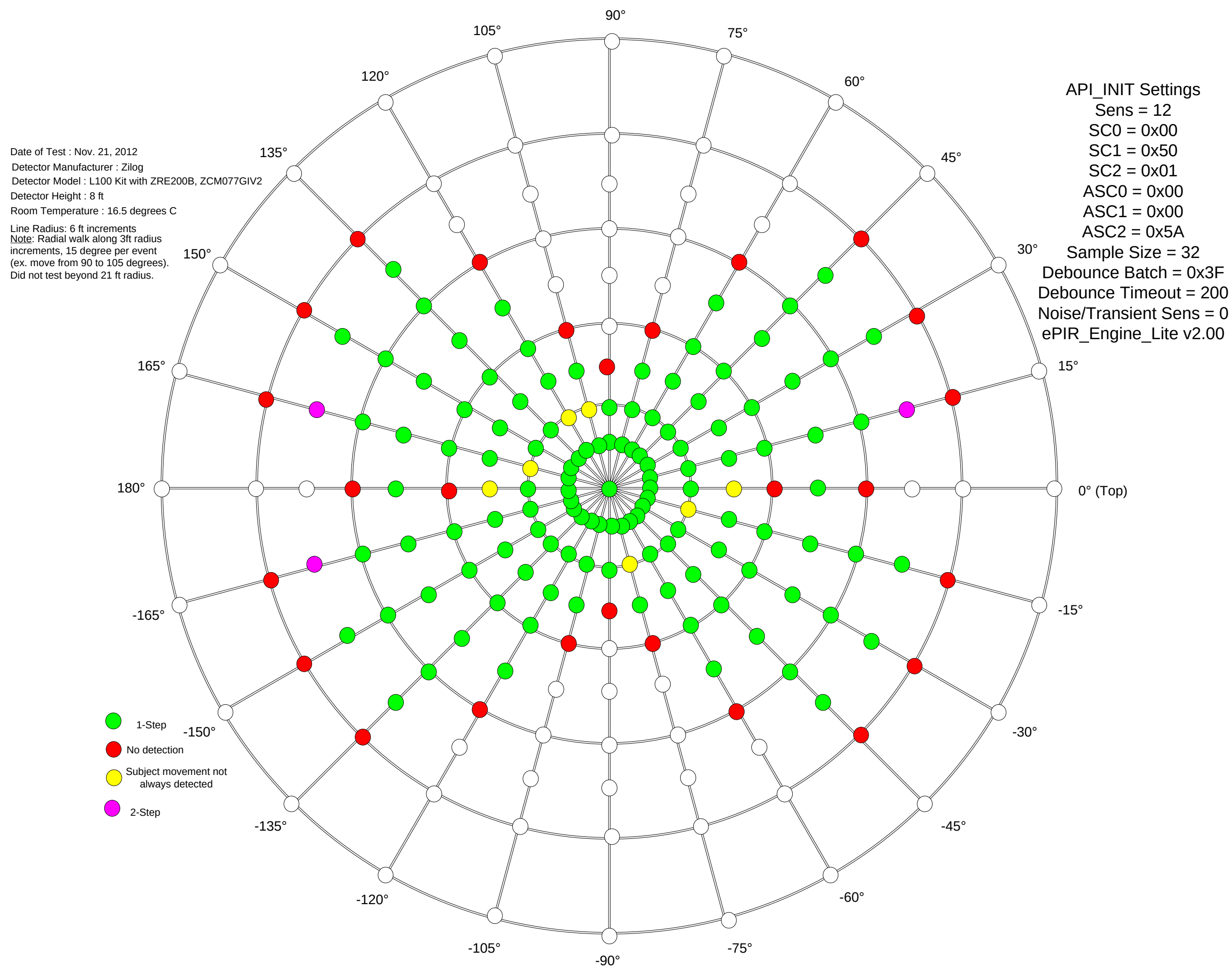

Figure 14. ZCM077GIV2 with ZRE200GE (8-Foot Ceiling)

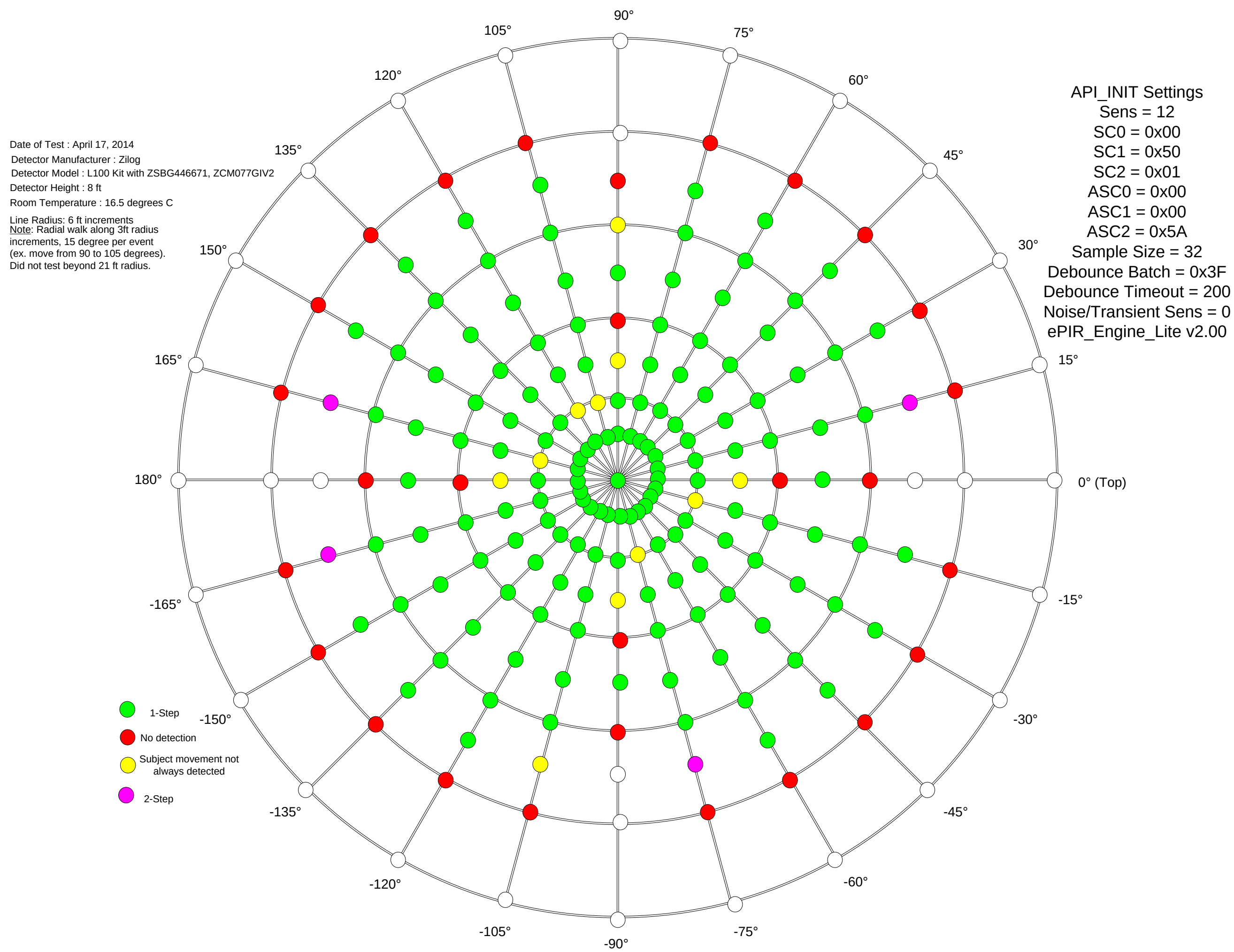

Figure 15. ZCM077GIV2 with ZSBG446671 (8-Foot Ceiling)

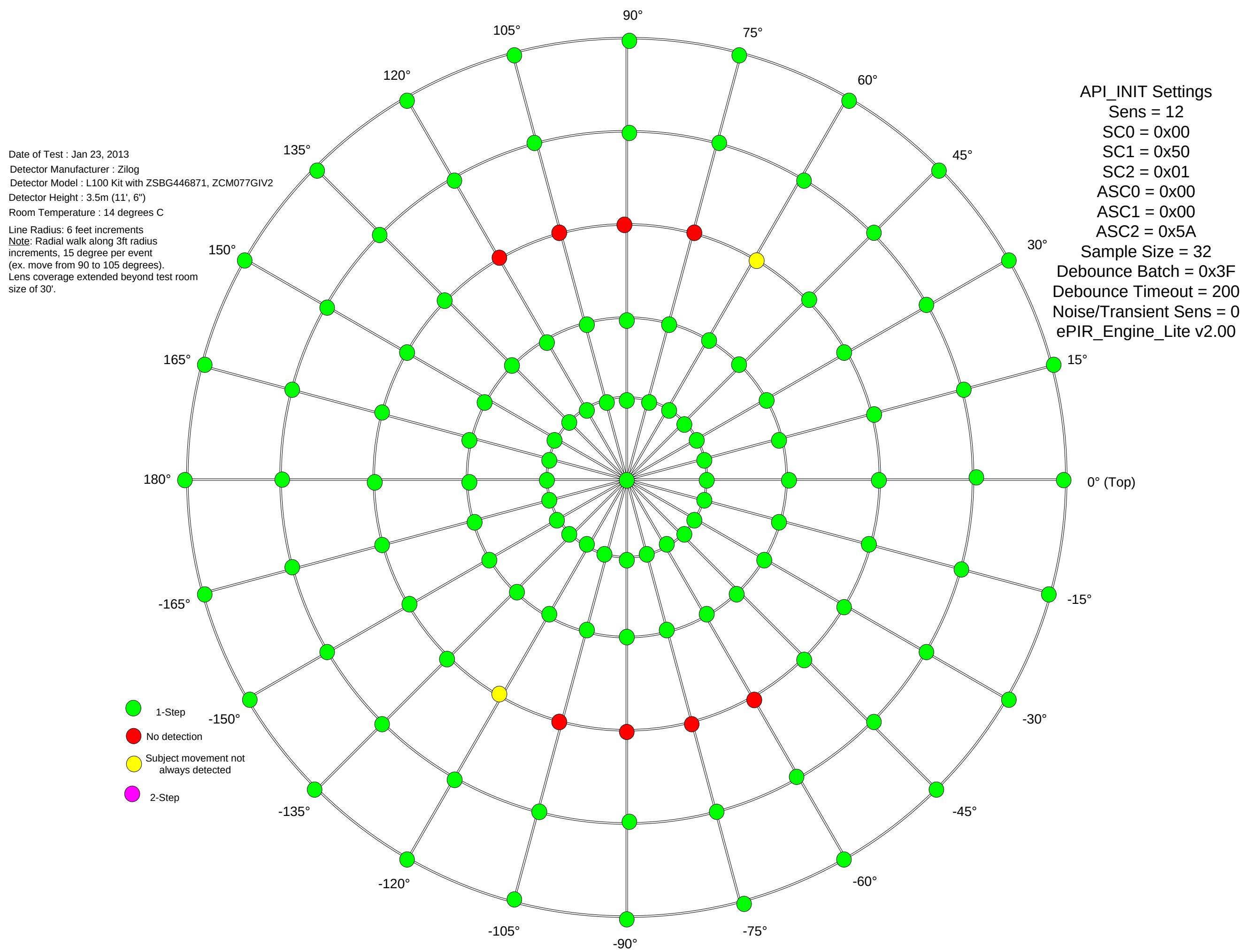

Figure 16. ZCM077GIV2 with ZSBG446671 (11.5-Foot Ceiling)

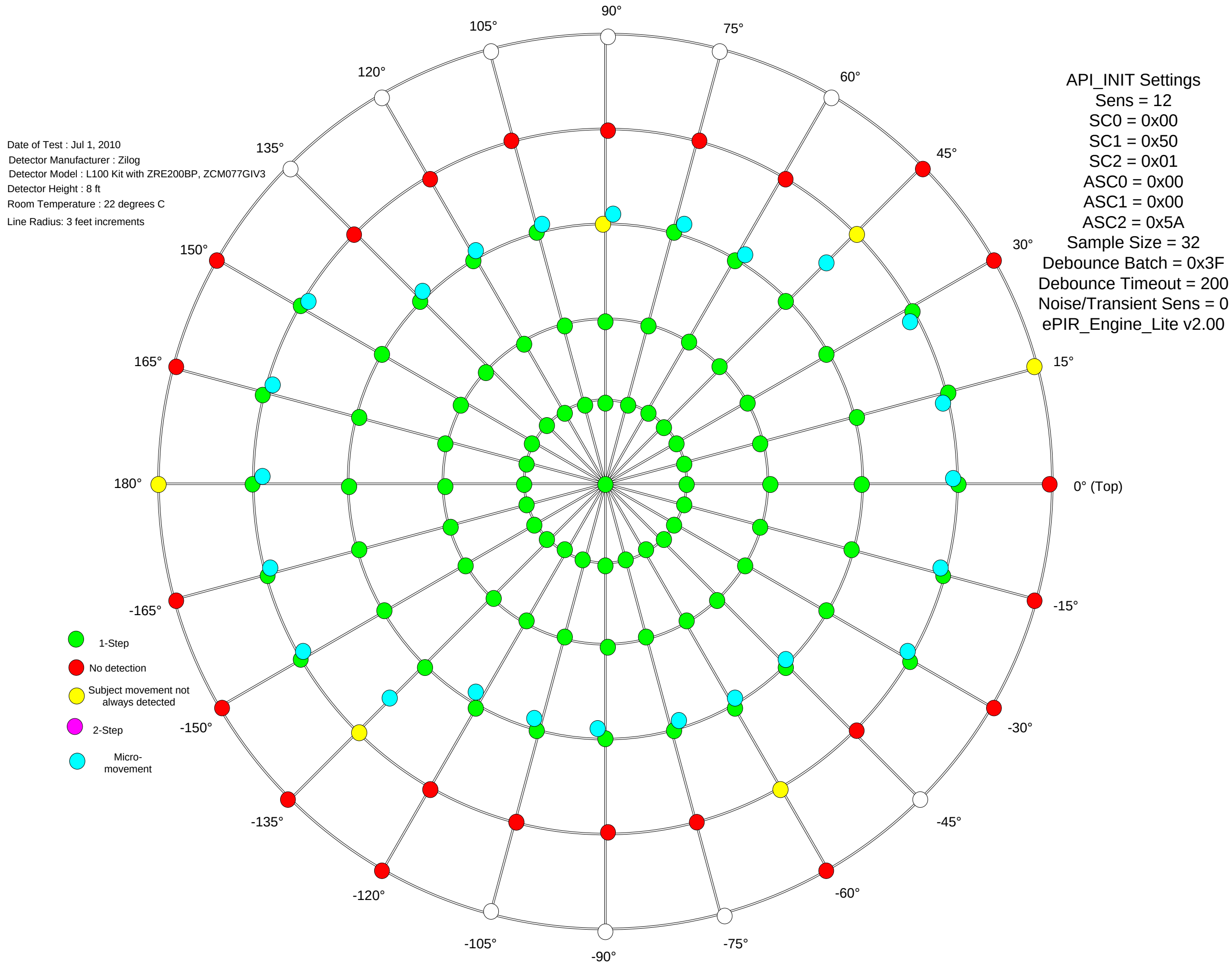

Figure 17. ZCM077GIV3 with ZRE200GE (8-Foot Ceiling)

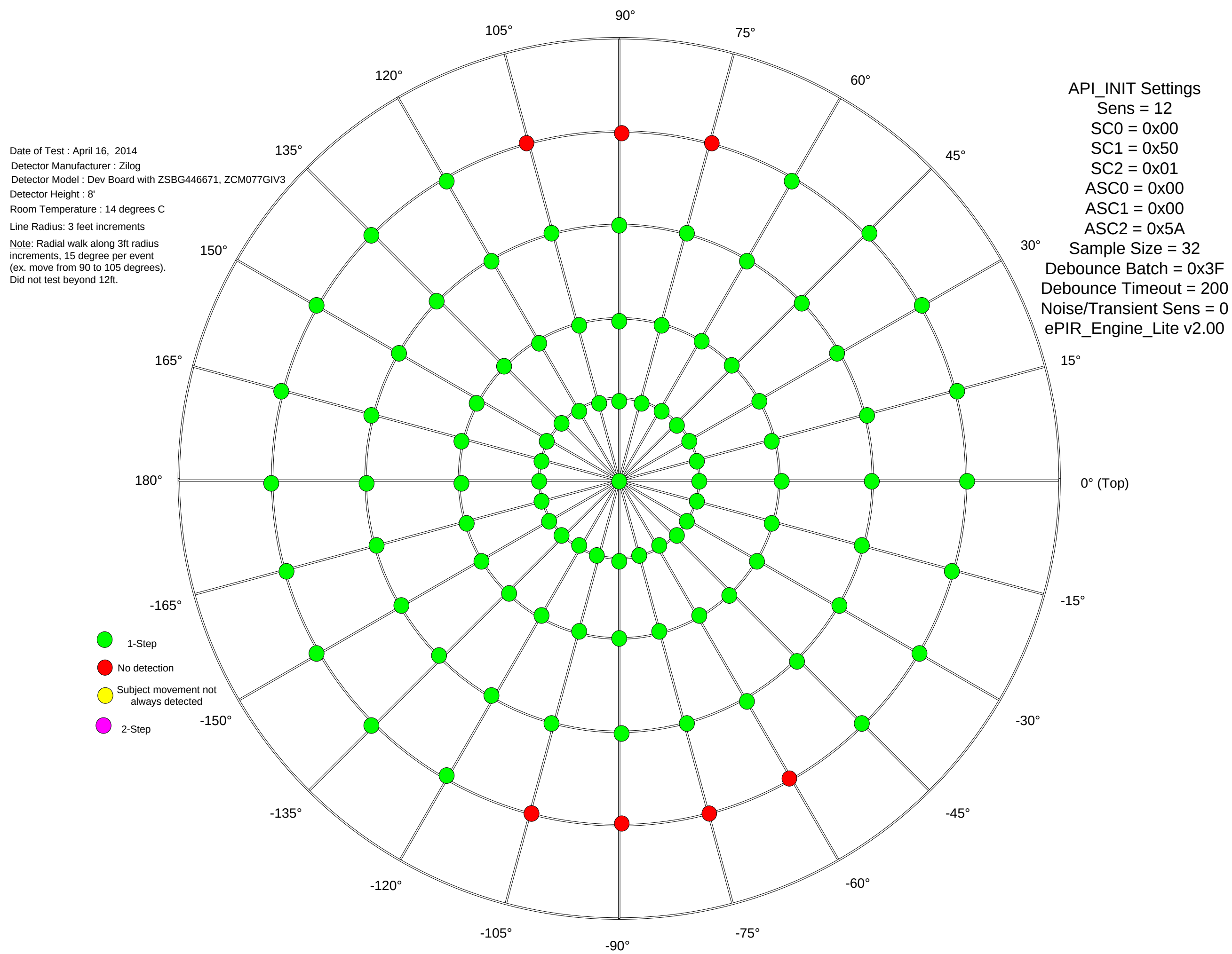

Figure 18. ZCM077GIV3 with ZSBG446671 (8-Foot Ceiling)

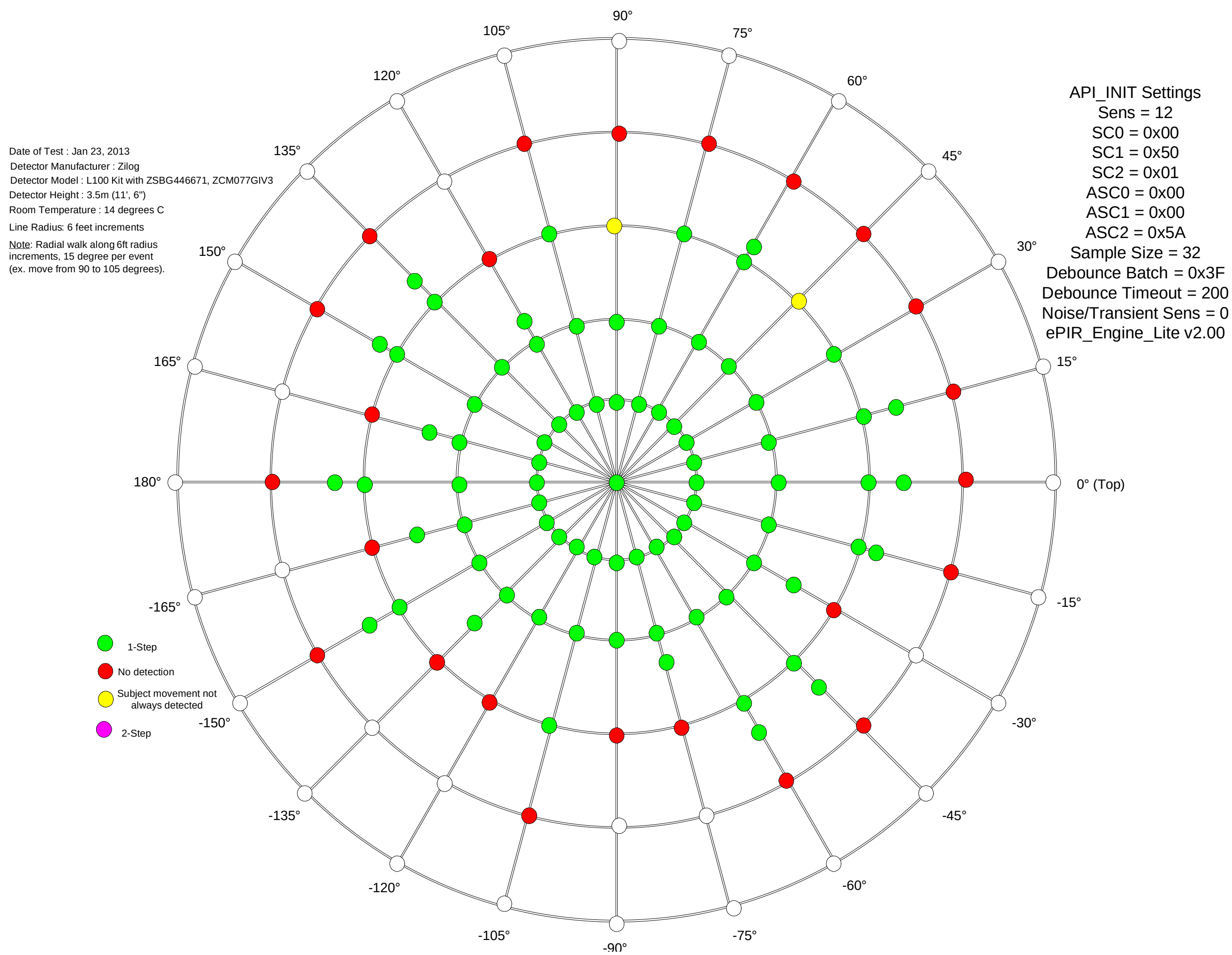

Figure 19. ZCM077GIV3 with ZSBG446671 (11.5-Foot Ceiling)

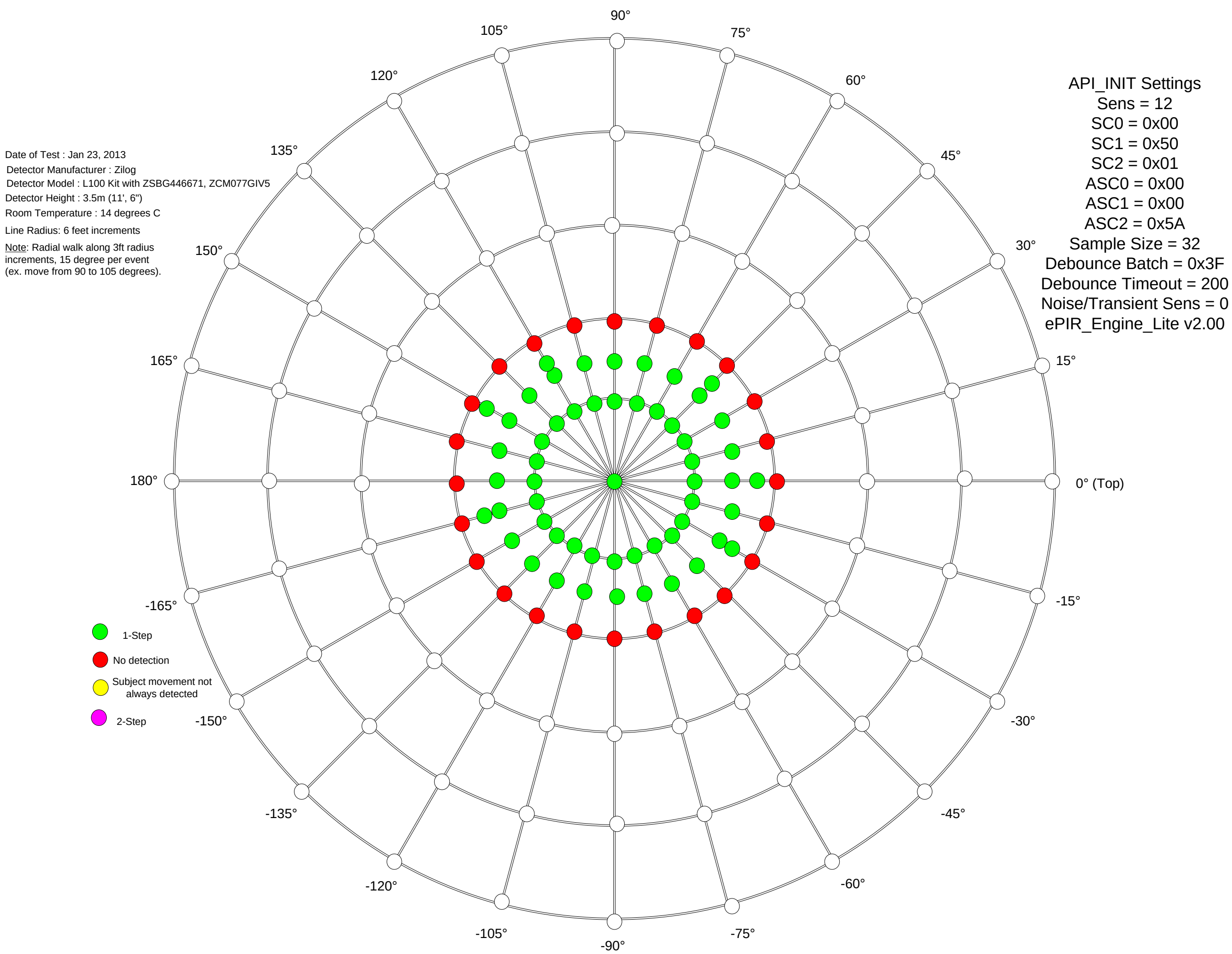

Figure 20. ZCM077GIV5 with ZSBG446671 (11.5-Foot Ceiling)

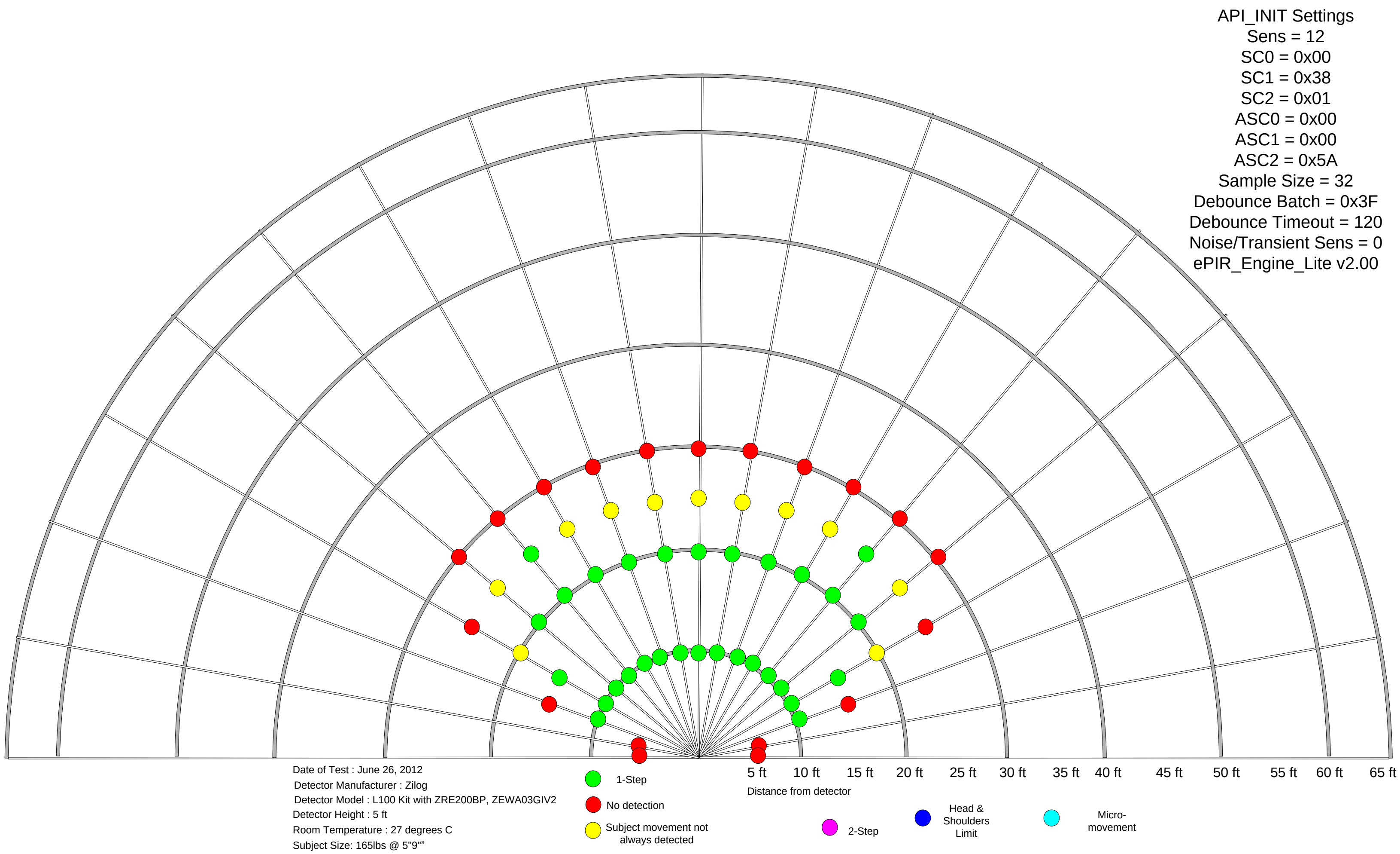

Figure 21. ZEWA03GIV2 with ZRE200GE

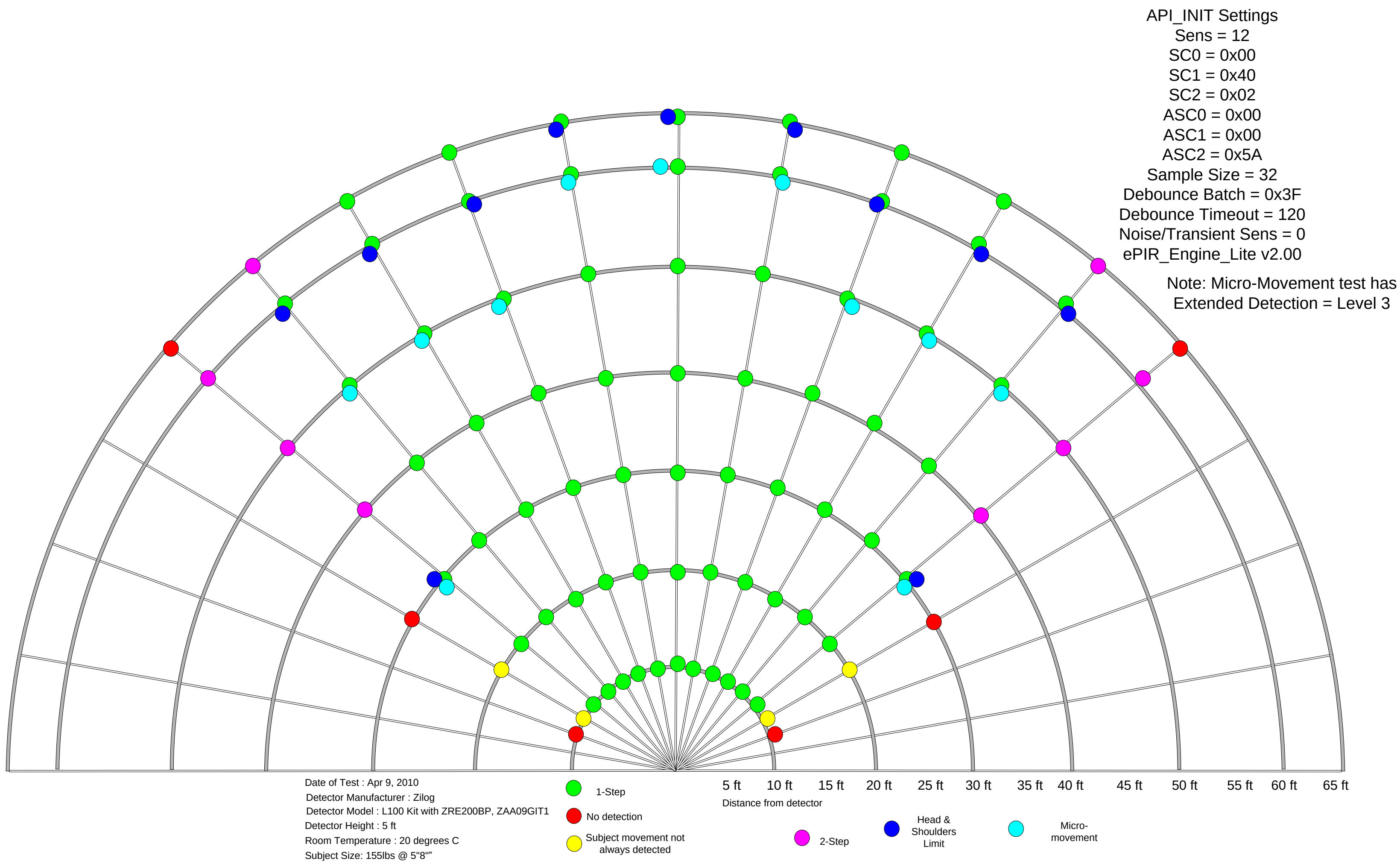

Figure 22. ZAA09GIT1 with ZRE200GE

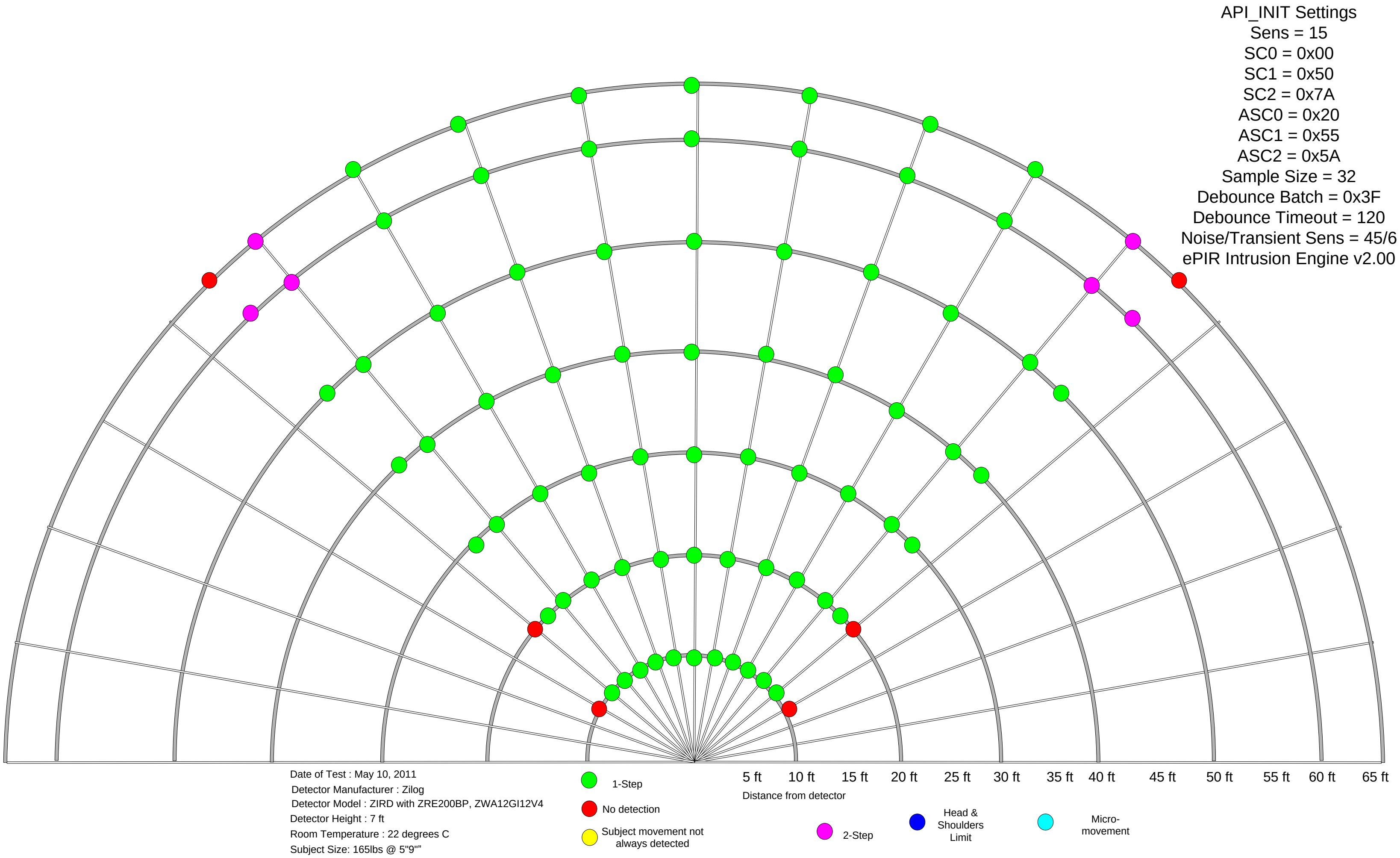

Figure 23. ZWA12GI12V4 with ZRE200GE

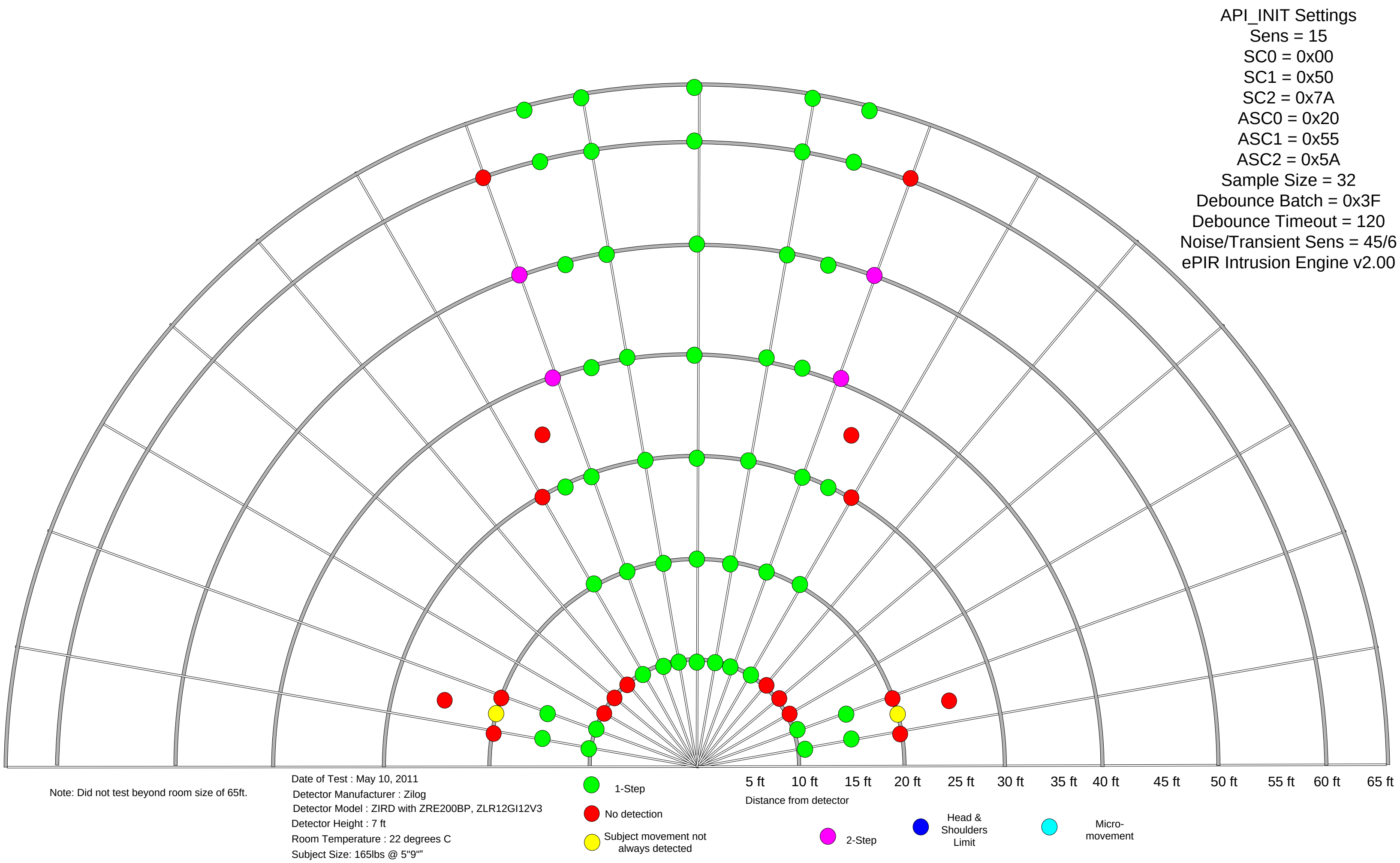

Figure 24. ZLR12GI12V3 with ZRE200GE

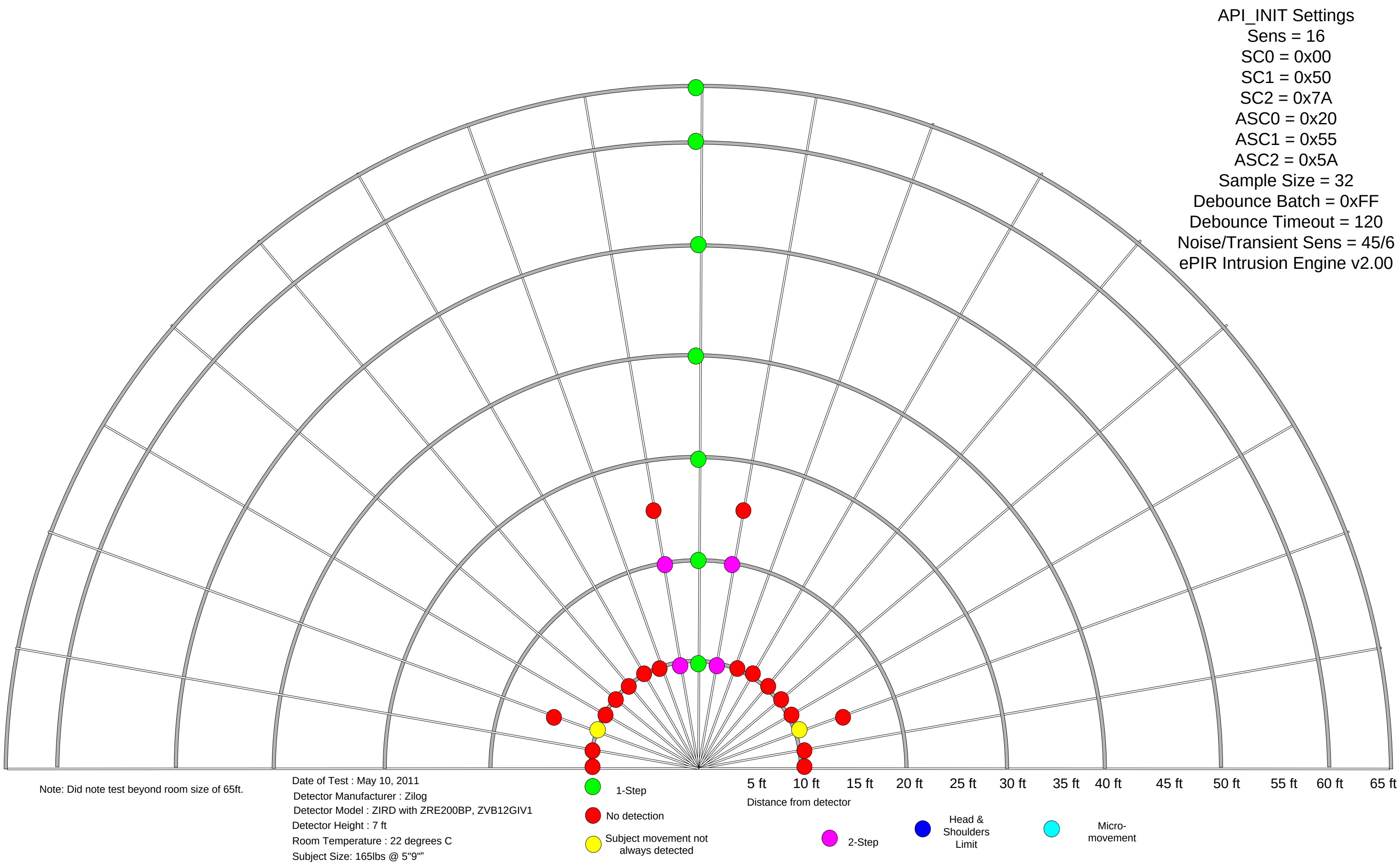

Figure 25. ZVB12GIV1 with ZRE200GE

Optimizing API Settings for a New Lens or Sensor

There are two main items to consider when evaluating a new lens and/or pyroelectric sensor: Performance and Stability. Performance refers to the ability of the system to detect valid motion and includes minor motion like moving heads or arms, major motion like a human body moving in to the detection area, range or distance and target speed. Stability refers to the system's ability to reject false motion events that can come from many sources including EMI, power supply noise, ambient temperature changes, PCB thermal drift, sunlight, air movement and pets. Determining the optimal API settings is an iterative process between system stability and system performance.

When evaluating a new lens or pyroelectric sensor not specifically covered in this document, start with proven settings from an `API_INIT_xx.h` file for a lens/sensor combination with similar characteristics. For example, if the new lens is a ceiling mount type with a wide area of coverage and long focal length, start with the API_INIT_02 settings for the ZCM077GIV3 lens using a quad sensor (like the ZSBDI46504AA). Avoid evaluating both a new sensor and new lens simultaneously. Introducing two unknowns can lead to confusion if there are detection issues. After both the new items have been verified and optimized independently, they can be combined and optimized as required.

Key API Settings

The following four primary parameters will have the greatest impact on performance:

- Sensitivity (ePIR_Sensitivity)
- Range (ePIR_SC2, bits 2–0)
- Frequency Response (ePIR_SC1, bits 6–3)
- Extended Detection (ePIR_SC0, bits 7–6)

Some guidelines on adjusting these parameters are given below. Adjust one parameter and retest for both performance and stability. If the results are still not satisfactory, restore that parameter and repeat with the next parameter. Continue until satisfactory results are found. Adjusting one parameter at a time can help in determining what parameter most significantly improved performance so only the necessary adjustments are required. This helps to ensure that stability is maintained.

Sensitivity. This parameter is one of the key ZMOTION Engine parameters. It acts as a motion signal 'duration' check. A smaller sensitivity value means the duration of a valid motion signal can be shorter. A larger sensitivity value means a valid motion signal's duration must be longer. To increase sensitivity, make this number smaller. To improve stability, make this number larger. Valid values are 0–254. In most applications values between 6 and 40 are typical, with 12 to 16 being the most common. As the numbers get smaller the effect is more pronounced. For example, changing from 10 to 8 may make a very noticeable difference. However, changing from 30 to 35 may not be noticeable at all. Keep this logarithmic property in mind when making adjustments.

Range. This parameter acts as a motion signal *amplitude* check. A smaller range value means the amplitude of a valid motion signal can be lower. A larger range value means the amplitude of a valid motion signal must be higher. The amplitude is relative to the average DC level of the sensor signal. To increase range, make this number smaller. To improve stability, make this number larger. Valid values are 0–7. In most applications values between 1 and 3 are typical, with 2 being the most common. As the number gets smaller the effect is more pronounced. For example, changing from 3 to 1 may make a very noticeable difference. However, changing from 5 to 7 may not be noticeable at all. Keep this *logarithmic* property in mind when making adjustments.

Frequency Response. This parameter acts as a motion signal *high-pass filter* check. A larger frequency response value means a valid motion signal's rate of change (i.e, slope) can be slower; i.e., the slope can have a longer time. A smaller frequency response value means a valid motion signal's rate of change (slope) must be faster. To increase detection performance, make this number larger – allowing lower speed targets to be detected. To improve stability, particularly to environmental temperature changes, make this number smaller. Valid values are `0x00` to `0x0F` (maximum value depends on window size). In most applications values between `0x04` and `0x0C` are typical, with `0x08` to `0x0A` being the most common.

Extended Detection. This parameter acts as a secondary motion detection check that runs in parallel with the normal detection check. The Extended Detection parameter looks for signals that extend beyond the typical parameters. Examples are very fast-moving targets (generating very small, high frequency motion signals) and very slow-moving targets (generating very small low frequency motion signals). This parameter also has the effect of improving detection of minor motion events (i.e., *micro-motion*). However, these types of events are also common signatures of false motion events, so extended detection must be used carefully. Valid values are 0–3 with 0 providing the lowest sensitivity (not off) and 3 providing the highest sensitivity. In most applications, values of 0–2 are typical, with 0 or 1 being the most common.

Increasing the value of this parameter increases the probability of false motion detection events. To offset this effect, the parameter can be kept at a low value (or disabled) and increased temporarily after one or more motion events have been detected, verifying that the area is occupied. This method can help provide micro-motion detection when it is required, while providing good stability when the area is unoccupied.

Programming Tip

An Extended Detection value of 0 does not disable the Extended Detection; it is simply the lowest setting. A simple workaround to completely disable the Extended Detection feature is to turn on Directional Detection (EPS0_DIRECTION_CONTROL in EPIR_SC0). Note that the Motion Direction indication bit (EPS0_DIRECTION) in EPIR_SC0 can be ignored and all detected motion events will still be indicated via the EPS0_MOTION_DETECTED bit.

Additional API Settings

In most cases adjusting the key API settings listed above are all that is required for achieving acceptable stability and performance. However, in some cases fine tuning of additional parameters may be necessary. The following API settings can be used to fine-tune motion detection stability and performance.

Lock Level. This parameter acts as an extension of the Range setting. It automatically adapts to spurious noise in the motion signal and ensures that a motion event has the proper profile. A smaller Lock Level value means a valid motion signal's amplitude can be lower. A larger Lock Level value means a valid motion signal's amplitude must be higher. To increase range, make this number smaller. To increase stability, make this number larger. Valid values are 0–7. In most applications values between 1 and 3 are typical, with 2 being the most common.

Debounce Batch. This parameter controls the amount of 'glitch' filtering applied to the motion signal. It filters out small amounts of spurious noise on the motion signal, preventing them from initiating a motion event. The parameter is implemented as a bit mask with valid values of `0x01`, `0x03`, `0x07`, `0x0F`, `0x1F`, `0x3F`, `0x7F`, and `0xFF`. This provides a range of noise filtering from low (`0x01`) to high (`0xFF`).

Transient Sensitivity. This parameter controls the sensitivity of the transient detector. The transient detector monitors the motion signal for sudden dramatic changes in the signal and prevents them from causing false motion events. Sources of transient events can be EMI from nearby radio transmitters or other circuitry on the PCB. Valid values are 0–100, with 0 disabling the Transient Detector. Typical values are from 6 to 40. Lower values provide more protection from transient events at the cost of potentially rejecting larger motion signals. Choose a value high enough to allow all desired motion events to pass through while still rejecting unwanted transient events. When a transient event is detected bit 0 in ePIR_ASC0 (EM Transient Detected) is set. The application can read this during testing to determine the correct sensitivity level.

Noise Sensitivity. This parameter controls the sensitivity of the noise detector. The noise detector monitors the motion signal for random broadband noise and prevents it from causing false motions events. Sources of broadband noise can be EMI from nearby radio transmitters or other circuitry on the PCB. Valid values are determined by the window size selected, with 0 disabling the Noise Detector. When a noise event is detected bit 1 in ePIR_ASC0 (EM Noise Detected) is set. The application can read this during testing to determine the correct sensitivity level.

Typical values are defined in the following table.

Table 3. Noise Sensitivity API Settings

Window Size	Max PIR Noise Sensitivity Value	Typical Value
Small	12	8
Medium	29	18
Large	70	45

Lower values provide more protection from noise events at the cost of potentially rejecting smaller motion signals. Choose a value high enough to allow all desired motion events to pass through while still rejecting unwanted noise events.

System Stability

The first step is to ensure that the reference settings provide a stable starting point with your particular hardware design. Test for stability by setting up the system in an area with no motion activity and leaving for long periods of time.

The area or space used is based on your requirements, and ideally is an unoccupied space with conditions similar to the environment where the product will be installed. Alternatives are small unoccupied rooms such as a storage area or closet.

The amount of stability testing time is also based on your own requirements, and can be one to several days.

For the initial stability check, it may not be necessary to run for an extended duration or in a realistic representative environment. An hour or two may be enough to move forward with initial performance tests. As the settings are refined, the stability testing can be extended to get a better indication of system stability.

Stability issues can be introduced from a variety of sources. The major sources are:

- Temperature changes affecting the pyro sensor itself, from ambient and/or on-board sources
- Undesired signal sources in the coverage area, including visible HVAC vents, sunlight through a window, pets, etc.
- Electrical interference causing a perceived motion signal on the pyro sensor; sources could be nearby radio transmitters, local oscillators, and electrical wires

Adjusting the API settings is often different for each particular source of stability issues. For this reason, determining the source of the stability problem is essential to quickly resolving it. The following are some suggestions on how to address these issues.

Eliminating temperature-related stability issues involves isolating the pyro sensor from ambient and on-board heat sources. Having a well-sealed enclosure along with spacing the sensor away from heat-generating components is recommended.

The key API setting for reducing thermal oversensitivity is Frequency Response. Additional effective settings are Extended Detection and Range.

Eliminating undesired signal sources is not always an option. Often the motion detector must tolerate these sources as they are an integral part of the environment. Device placement within the space can improve stability. Ideal placement would keep sunlit areas and HVAC vents out of the coverage area or further from the detector.

The key API setting for reducing sensitivity to pets and other undesired signal sources is Frequency Response. Additional effective settings are Sensitivity and Lock Level.

For EMC related events, the Transient and Noise Detection registers can be adjusted to eliminate the false trigger.

System Performance

After you have a known stable system (either long-term stability or just a quick initial check), it is time to test the performance. Referring to an existing coverage map or specification of the new lens can be very helpful. Using the coverage map provided by the lens manufacturer, perform a cursory walk test to determine if the lens is at or near its intended performance. If the lens is performing at or better than specifications, continue walk testing to determine if results are satisfactory for the application.

If the results are not satisfactory, the first place to look for possible problems is lens position relative to the sensor. Check that the lens is positioned correctly over the sensor as per the lens manufacturer's specification and that the focal length (distance from the lens to the sensor elements) is correct. Minor variations in position can have significant effects on test results. Correct any discrepancies and retest if required.

After the physical lens position has been verified, and if performance is still not satisfactory, the next step is to adjust API settings. *See section 6 above*.

After the optimum walk test performance has been obtained, the stability must be verified. This test ensures that the API settings do not result in false-positive motion events. The test should be performed in an area where actual motion activity will not occur during the test period.

A note on new sensor testing: In general, the variation between different sensors is small and therefore not much is required to adjust for it. Due to this, most of the recommended API settings are tied primarily to lens types. In most cases, using the unchanged API settings for the same lens on a different sensor is possible. Only when moving between major sensor types (ex, from a dual-element to a quad-element or vice-versa) is it possible that some changes need to be made.

For ZMOTION Intrusion applications, the procedure is very similar. In general, an intrusion application may require less aggressive settings to further reduce the risk of false motion events. Additionally, note that there are minor differences in the API settings compared to ZMOTION Detection and Control.

To learn more about the ZMOTION API registers, refer to the ZMOTION Detection and Control Product Specification (PS0285) and to the ZMOTION Intrusion Detection Product Specification (PS0288).

The following sequence determines the best settings for a new lens or pyroelectric sensor:

1. Identify the closest matching lens and associated API_INIT_xx file.
2. Compile your project using these API settings.
3. Perform stability test.
4. If stability is OK, skip to Step 6.
5. Adjust API settings (as described in section 6 and 7) and go to step 3.
6. Perform initial basic walk test using the lens manufacturers coverage map.
7. If performance is OK, repeat the stability test.
8. If stability is not OK, return to Step 5 to adjust the API settings.
9. If stability is OK, the API settings have been found and more in depth testing can be performed if desired.

Summary

Zilog's ZMOTION Detection and Control solution provides a flexible method for optimizing a variety of lighting and occupancy applications. Zilog uses popular, high quality lens and pyroelectric sensor combinations, along with proven API settings, to establish quick development times to meet our customer's requirements.

Appendix A. Sample API INIT Header File

The following example code lists the `API_INIT_02.h` API initialization header file.

```
/*****************************************************************************/
//  Copyright (C) 1999-2014 Zilog, Inc.
//  All Rights Reserved.
//
//  File: API_INIT_02.h
//
//  Supported Lenses:       Fresnel Technologies CM 0.77 GI V3
//  Supported PIR Sensors:  Nicera SBG446-671, RE200B-P, SBG323-671
//
//  This file contains the PIR Engine configuration parameters for the lens
//  and PIR combinations identified above. It should be included in the ZMOTION
//  application project along with ePIR_API.h and ePIR_API.c and loaded into
//  the API registers during initialization.
//
//  Refer to ZMOTION Engine API Lens and Sensor Configuration Guide (WP0018)
//  For more information about configuring these settings.
//
//  See the "ZMotion_Basic" sample project for a full example of creating a
//  ZMOTION project.
//
//  Revision: 1.0 - Initial Release
//  Revision: 1.1 - Settings adjustments, format changes
/*****************************************************************************/


#ifndef API_INIT_H
#define API_INIT_H

//////////////////////////////////////////////////////////////////////////
// API configuration for Normal Scan Rate Mode
//////////////////////////////////////////////////////////////////////////

//////////////////////////////////////////////////////////////////////////
// ePIR_Sensitivity Register Default Setting for Normal Scan Rate
// Lower values provide greater sensitivity
//////////////////////////////////////////////////////////////////////////
#define EPIR_SENSITIVITY_DEF  12

//////////////////////////////////////////////////////////////////////////
// ePIR_SC0 Default Configuration for Normal Scan Rate
//////////////////////////////////////////////////////////////////////////
// Bit 6-7 - Extended Detection - Level 0        (00)
// Bit 5 - Engine Disabled - Engine Controlled  (0)
```

```
// Bit 4 - MD Suspend - OFF                      (0)
// Bit 3 - Motion Direction Control - OFF        (0)
// Bit 2 - Motion Direction - Engine Controlled (0)
// Bit 1 - Motion Detected - Engine Controlled  (0)
// Bit 0 - PIR Stable - Engine Controlled       (0)
#define EPIR_SC0_DEF  0x00

//////////////////////////////////////////////////////////////////////////
// ePIR_SC1 Default Configuration for Normal Scan Rate
//////////////////////////////////////////////////////////////////////////
// Bit 7 - Engine Timer Tick                     (0)
// Bits 6-3 - Frequency Response                 (1010)
// Bit 2 - PIR Scan Rate - Normal Mode           (0)
// Bit 1 - Reserved                              (0)
// Bit 0 - Dual Pyro Enable - OFF                (0)
#define EPIR_SC1_DEF  0x50

//////////////////////////////////////////////////////////////////////////
// ePIR_SC2 Default Configuration for Normal Scan Rate
// Lower values provide greater range
//////////////////////////////////////////////////////////////////////////
// Bits 7-3 - Reserved                           (00000)
// Bits 2-0 - Range                              (010)
#define EPIR_SC2_DEF  0x01

//////////////////////////////////////////////////////////////////////////
// ePIR_SC3 Default Configuration for Normal Scan Rate
//////////////////////////////////////////////////////////////////////////
// Bits 7-0 - ANAx Scan Request - None           (00000000)
// No ADC Scan requests made during Init
#define EPIR_SC3_DEF  0x00

//////////////////////////////////////////////////////////////////////////
// ePIR_ASC0 Default Configuration for Normal Scan Rate
//////////////////////////////////////////////////////////////////////////
// Bits 7-5 - Reserved                           (000)
// Bit 4 - Buffer Refresh - OFF                  (0)
// Bit 3 - New Sample - Engine Controlled        (0)
// Bit 2 - MD Origin - Engine Controlled         (0)
// Bit 1 - EM Noise - Engine Controlled          (0)
// Bit 0 - EM Transient - Engine Controlled      (0)
#define EPIR_ASC0_DEF  0x00

//////////////////////////////////////////////////////////////////////////
// ePIR_ASC2 Default Configuration for Normal Scan Rate
//////////////////////////////////////////////////////////////////////////
// Bits 7-5 - Lock Level - 2                     (010)
// Bits 4-3 - Window Size - 3                    (11)
```

```
// Bits 2-0 - Window Update Rate - 2            (010)
#define EPIR_ASC2_DEF  0x5A

//////////////////////////////////////////////////////////////////////////
// ePIR_Sample_Size Default Value for Normal Scan Rate
//////////////////////////////////////////////////////////////////////////
// ePIR Sample Size - 32
#define EPIR_SAMPLE_SIZE_DEF  32

//////////////////////////////////////////////////////////////////////////
// ePIR_Debounce Time Default Value for Normal Scan Rate
//////////////////////////////////////////////////////////////////////////
// ePIR Debounce Time - 200
#define EPIR_DEBOUNCE_DEF  200

//////////////////////////////////////////////////////////////////////////
// ePIR_Debounce_Batch Default Value for Normal Scan Rate
//////////////////////////////////////////////////////////////////////////
// ePIR Debounce Batch Size - 3F                (00111111)
#define EPIR_DEBOUNCE_BATCH_DEF  0x3F

//////////////////////////////////////////////////////////////////////////
// ePIR_Transient_Sense Default Value for Normal Scan Rate
//////////////////////////////////////////////////////////////////////////
// ePIR Transient Sensitivity - Disabled
#define EPIR_TRANSIENT_SENSE_DEF  0x00

//////////////////////////////////////////////////////////////////////////
// ePIR_Noise_Sense Default Value for Normal Scan Rate
//////////////////////////////////////////////////////////////////////////
// ePIR Noise Sensitivity - Disabled
#define EPIR_NOISE_SENSE_DEF  0x00

#endif?
```

Appendix B. Sample API Initialization Code (main.c)

The following code listing presents the ZMOTION Engine API Initialization copy-down prior to calling the EPIR_INIT macro function.

```
…
…
///////////////////////////////////////////////////////////
// ePIR Engine Initialization
// Initialize ePIR API before calling EPIR_INIT Macro
// Initialization values are defined in API_INIT_xx.h
///////////////////////////////////////////////////////////
    ePIR_Sensitivity = EPIR_SENSITIVITY_DEF;
    ePIR_SC0 = EPIR_SC0_DEF;
    ePIR_SC1 = EPIR_SC1_DEF;
    ePIR_SC2 = EPIR_SC2_DEF;
    ePIR_SC3 = EPIR_SC3_DEF;

    ePIR_ASC0 = EPIR_ASC0_DEF;
    ePIR_ASC2 = EPIR_ASC2_DEF;
    ePIR_Sample_Size = EPIR_SAMPLE_SIZE_DEF;
    ePIR_Debounce_Batch = EPIR_DEBOUNCE_BATCH_DEF;
    ePIR_Debounce = EPIR_DEBOUNCE_DEF;
    ePIR_Noise_Sense = EPIR_NOISE_SENSE_DEF;
    ePIR_Transient_Sense = EPIR_TRANSIENT_SENSE_DEF;

    ePIR_Enable = EPIR_ENABLE_PATTERN;  // Allow the Engine to be enabled

// This Macro calls the initialization routine in the Engine - it starts the
// Motion Detection Engine. It must be executed once after the Engine registers
// are initialized (above).
    EPIR_INIT;
…
…?
```

> **Warning:** DO NOT USE THIS PRODUCT IN LIFE SUPPORT SYSTEMS.

LIFE SUPPORT POLICY

ZILOG'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS PRIOR WRITTEN APPROVAL OF THE PRESIDENT AND GENERAL COUNSEL OF ZILOG CORPORATION.

As used herein

Life support devices or systems are devices which (a) are intended for surgical implant into the body, or (b) support or sustain life and whose failure to perform when properly used in accordance with instructions for use provided in the labeling can be reasonably expected to result in a significant injury to the user. A critical component is any component in a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system or to affect its safety or effectiveness.

Document Disclaimer

©2014 Zilog Inc. All rights reserved. Information in this publication concerning the devices, applications, or technology described is intended to suggest possible uses and may be superseded. ZILOG, INC. DOES NOT ASSUME LIABILITY FOR OR PROVIDE A REPRESENTATION OF ACCURACY OF THE INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED IN THIS DOCUMENT. ZILOG ALSO DOES NOT ASSUME LIABILITY FOR INTELLECTUAL PROPERTY INFRINGEMENT RELATED IN ANY MANNER TO USE OF INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED HEREIN OR OTHERWISE. The information contained within this document has been verified according to the general principles of electrical and mechanical engineering.

Z8 Encore!, Z8 Encore! XP and ZMOTION are trademarks or registered trademarks of Zilog, Inc. All other product or service names are the property of their respective owners.